Improvement of DCT-based Compression Algorithms Using Poisson's Equation

Katsu Yamatani* and Naoki Saito, *Senior Member, IEEE*

Abstract

We propose two new image compression-decompression methods that reproduce images with better visual fidelity, less blocking artifacts, and better PSNR, particularly in low bit rates, than those processed by the JPEG Baseline method at the same bit rates. The additional computational cost is small, i.e., linearly proportional to the number of pixels in an input image. The first method, the "full mode" *polyharmonic local cosine transform* (PHLCT), modifies the encoder and decoder parts of the JPEG Baseline method. The goal of the full mode PHLCT is to reduce the code size in the encoding part and reduce the blocking artifacts in the decoder part. The second one, the "partial mode" PHLCT (or PPHLCT for short), modifies only the decoder part, and consequently, accepts the JPEG files, yet decompresses them with higher quality with less blocking artifacts. The key idea behind these algorithms is a decomposition of each image block into a *polyharmonic* component and a *residual*. The polyharmonic component in this paper is an approximate solution to Poisson's equation with the Neumann boundary condition, which means that it is a smooth predictor of the original image block only using the image gradient information across the block boundary. Thus the residual—obtained by removing the polyharmonic component from the original image block—has approximately zero gradient across the block boundary, which gives rise to the fast-decaying DCT coefficients, which in turn lead to more efficient compression-decompression algorithms for the same bit rates. We show that the polyharmonic component of each block can be estimated solely by the first column and row of the DCT coefficient matrix of that block and those of its adjacent blocks and can predict an original image data better than some of the other AC prediction methods previously proposed. Our numerical experiments objectively and subjectively demonstrate the superiority of PHLCT over the JPEG Baseline method and the improvement of the JPEG-compressed images when decompressed by PPHLCT.

K. Yamatani is with the Department of Systems Engineering, Shizuoka University, 3-5-1 Johoku, Hamamatsu, Shizuoka 432-8561 Japan (e-mail: yamatani@sys.eng.shizuoka.ac.jp, tel/fax:+81-53-478-1220).

N. Saito is with the Department of Mathematics, University of California, Davis, One Shields Avenue, Davis, CA 95616-8633 USA (e-mail: saito@math.ucdavis.edu, tel:530-754-2121 fax:530-752-6635).

EDICS Category: 1-STIL, 1-STAN, 2-NFLT, 2-REST

I. INTRODUCTION

Currently, the most popular image compression scheme is the JPEG standard for still images and the MPEG standard for video sequences. Although new standards such as JPEG-2000 and MPEG-21 (both of them have adopted wavelets) are emerging, many systems including digital cameras and image databases still use the JPEG standard [1], [2], which is based on the *Discrete Cosine Transform* (DCT) [3], [4], [5]. The reasons why DCT—sometimes called DCT Type II or DCT-II [4], [5]—is used in many compression algorithms instead of the Discrete Fourier Transform (DFT) or the Discrete Sine Transform (DST) are the following.

1) DCT theoretically approximates the Karhunen-Loève Transform (KLT) if we assume the first order Markov property in a given set of signals to be compressed [6]. It is well known that KLT is the most efficient *linear* transform for a collection of signals if the reconstruction error is measured in the mean square error (MSE) [6], [7].

2) Even more important is its treatment of the block boundaries. Taking DCT of an image block is equivalent to: a) extending the data by even reflection at the block boundaries; b) taking DFT of the extended data by viewing it as periodic with the period twice as long as that of the original block; and c) discarding $3/4$ of the resulting coefficients that were created by the even reflection and extension. This implies that DCT views the data as a *continuous* function across the block boundaries as long as there is no intrinsic singularity within the block. This should be contrasted with DFT and DST: they view the data as *discontinuous* across the block boundaries *even if there is no intrinsic singularity within the block*. This leads to the faster decay of the DCT coefficients than the DFT or DST coefficients, which is a desirable property in any transform coding system because it allows us to approximate the original data with a fewer number of terms in the series expansion used in the transform. More precisely, let $F_{\boldsymbol{k}}$ be the $\boldsymbol{k}$th DCT coefficient of an image block, where $\boldsymbol{k} = (k_1, k_2)$ is a pair of indices specifying the spatial frequencies along vertical and horizontal directions. Then, one can show that $|F_{\boldsymbol{k}}| \leq C\|\boldsymbol{k}\|^{-2} = C/(k_1^2 + k_2^2)$ for some constant $C > 0$ (see also Section II). We denote this by $F_{\boldsymbol{k}} = O(\|\boldsymbol{k}\|^{-2})$ in this paper. On the other hand, the magnitude of the DFT and DST coefficients of the same block is of $O(\|\boldsymbol{k}\|^{-1})$, i.e., those decay slower than the DCT coefficients. See Theorem 1 in Section II for the precise statement relating the smoothness of a function and the decay rate of its Fourier coefficients.

3) The lowest frequency basis vector of DCT is completely flat, i.e., truly measuring the important

"DC" component of an image block, which is not the case in DST and in the other versions of DCT called DCT Type I, III, and IV (see e.g., [4], [5]).

Although DCT-based codecs, such as the JPEG Baseline sequential codec, are quite popular, there are two problems in them that we want to overcome. First, the decay rate $O(\|\boldsymbol{k}\|^{-2})$ of the DCT coefficients is still too slow. We would like to make the transformed coefficients decay faster in order to achieve better compression. Second, such codecs generate the infamous blocking artifacts (visible discontinuities between adjacent blocks) when the compressed data—especially the ones with low bit rates—are decompressed, which are quite noticeable and annoying for human observers. Many techniques have been developed to remove the blocking artifacts while keeping the objectively measured image quality (e.g., PSNR) constant, e.g., [8], [9], [10], [11], [12]. These all operate on the spatial domain, i.e., these are post-processing techniques after the decompression is done. On the other hand, another set of techniques has been suggested as an option in the JPEG standard [1, Sec. 16.1] and further explored in [13], [14], [15]. These methods are all based on the prediction of the AC coefficients by fitting quadratic or cubic polynomial surface using the DC coefficients of several blocks. In general, polynomial surface fitting techniques are effective to remove the blocking artifacts especially for images compressed in low bit rates. However, for the actual implementation of these techniques, there is a common difficulty to distinguish intrinsic singularities in the original image from the artificial discontinuities that cause the blocking artifacts.

To address these problems, Saito and Remy recently developed a new transform called *Polyharmonic Local Sine Transform* (PHLST) [16], [17]. PHLST decomposes the data in each block into two components: the *polyharmonic component* and the *residual*. The polyharmonic component is obtained by solving the polyharmonic equation (e.g., Laplace's equation, biharmonic equation, etc.) given the boundary values (i.e., the pixel values along the block border) possibly with the estimates of derivatives across the boundary. Once this component is obtained, this is subtracted from the image block data to obtain the residual, whose Fourier sine series expansion has quickly decaying coefficients since the boundary values of the residual (possibly with their normal derivatives) vanish. If we use Laplace's equation as the polyharmonic equation in PHLST, then the DST coefficients of the residual component decay as $O(\|\boldsymbol{k}\|^{-3})$, which is faster than the DCT coefficients of the original block do. See Section II-C for more about PHLST. Although PHLST is very promising in certain applications such as local feature computation, directional derivatives estimation, image interpolation, and image zooming, it cannot directly and fully utilize the infrastructure of the JPEG Baseline method due to its use of DST instead of DCT as well as its storage

requirement of the block boundary information.

In this paper, we propose the *Polyharmonic Local Cosine Transform* (PHLCT) that compensates these problems of PHLST: it gives us the residual coefficients decaying as $O(\|\boldsymbol{k}\|^{-4})$, yet fully utilizes the JPEG Baseline infrastructure. There are two modes in PHLCT. One is the *full* mode that modifies the encoder and decoder parts of the JPEG Baseline procedure. The goal of the full mode PHLCT is to: 1) reduce the code size by encoding the DCT coefficients of the residual component that decay as $O(\|\boldsymbol{k}\|^{-4})$; and 2) reduce the blocking artifacts in the decoding process by the interpolation using the polyharmonic function. The other mode is what we call the *partial* mode. This mode modifies only the decoder part, and consequently it can take JPEG-compressed images and reproduce images with less blocking artifacts compared to those reproduced by the JPEG Baseline standard. Our methods can also be viewed as new and improved AC prediction methods over the previously proposed ones, e.g., [1, Sec. 16.1], [13], [14], [18], and [15].

The organization of this paper is as follows. In Section II, we set our notation and review the Fourier cosine series and PHLST. In Section III, we propose PHLCT and show why the Fourier cosine coefficients of the residual decay as $O(\|\boldsymbol{k}\|^{-4})$. Section IV describes how to compute the PHLCT representation of an input image from the DCT coefficients of that image. In Section V, we derive more practical approximation of the polyharmonic component than that in Section IV solely using the DC components of the current and adjacent blocks. Then we describe our compression-decompression algorithms using full and partial modes of PHLCT in Section VI. Section VII describes our numerical experiments. There we compare the performance of our methods in predicting the original AC coefficients with that of the other AC prediction methods mentioned above. Also, we quantify the image quality improvement of our methods over the JPEG Baseline method using several image quality measures. We finally conclude our paper in Section VIII with our future plan.

II. REVIEW OF FOURIER COSINE SERIES EXPANSION AND POLYHARMONIC LOCAL SINE TRANSFORM

In this section and the next, we shall deal with data in an arbitrary dimension $n \in \mathbb{N}$ since our method may be useful for compressing not only 2D images but also 3D data such as 3D seismic data. We shall then focus on the case of $n = 2$ for images starting Section IV.

A. Fourier series expansion and periodization

Let Ω be a block domain $\Omega = \{ \boldsymbol{x} \in \mathbb{R}^n | 0 < x_i < 1, i = 1, 2, \ldots, n \}$, where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$. Let us now consider a function f defined on the closed domain $\overline{\Omega} = \Omega \cup \partial\Omega$, where $\partial\Omega$ denotes the boundary of Ω. Then let us extend it periodically over the whole space $\mathbb{R}^n$, and let $\widetilde{f}$ be this periodic extension of f. We have the following basic theorem that relates the decay rate of the Fourier coefficients (i.e., the inner products between $\widetilde{f}$ and the complex exponentials) and the smoothness of the function $\widetilde{f}$. For the details and the proof, see Appendix of [17].

Theorem 1: Let f be a function defined on $\overline{\Omega} \in \mathbb{R}^n$ and $\widetilde{f}$ be its periodic extension to $\mathbb{R}^n$. Furthermore, let us assume that 1) $\widetilde{f} \in C^m(\mathbb{R}^n)$, where m is some nonnegative integer; and 2) $\partial^{m+1}\widetilde{f}/\partial x_i^{m+1}$, $i = 1, \ldots, n$, exists and of bounded variation in Ω. Then the Fourier coefficient $c_{\boldsymbol{k}}$ of f decays as $O(\|\boldsymbol{k}\|^{-m-2})$, $\boldsymbol{k} = (k_1, \ldots, k_n) \in \mathbb{Z}^n$, and $\|\boldsymbol{k}\|$ is the Euclidean (i.e., ℓ^2) norm of $\boldsymbol{k}$.

Note that if some head and tail of f do not match, i.e., unless we have $f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$ for all $i \in \{1, \ldots, n\}$ and all x_k with $0 \leq x_k \leq 1$, $k \neq i$, which is almost always violated in real data, then the periodic extension $\widetilde{f}$ becomes discontinuous. Thus its Fourier coefficients decay only as $O(\|\boldsymbol{k}\|^{-1})$ and reveal the annoying Gibbs phenomenon [19, Sec. 5].

B. Fourier cosine series expansion and even reflection

We now consider the Fourier cosine series expansion of a non-periodic function $f \in C^{m'}(\overline{\Omega})$ for some $m' \in \mathbb{N}$. Because of the equivalence of the Fourier cosine series expansion of f and the complex Fourier series expansion of the even-reflected version of f as briefly discussed in Introduction, the even-reflected version becomes at least continuous and periodic over the extended domain $[-1, 1]^n$. Thus, applying Theorem 1 with $m = 0$, the Fourier cosine coefficients of f decay as $O(\|\boldsymbol{k}\|^{-2})$. Unfortunately, even if f is sufficiently smooth on the block, i.e., $f \in C^{m'}(\overline{\Omega})$ with large positive m' including the case of $m' = \infty$, there is no guarantee that the even reflection preserves the continuity of the derivatives across the boundary. Therefore, the Fourier cosine coefficients cannot decay faster than $O(\|\boldsymbol{k}\|^{-2})$ in general.

C. Polyharmonic local sine transform

In order to have expansion coefficients decaying as $O(\|\boldsymbol{k}\|^{-3})$ or faster without generating the Gibbs oscillations, Saito and Remy proposed the *Polyharmonic Local Sine Transform* (PHLST) [16], [17]. PHLST also compensates several problems in the local trigonometric transforms (LTTs) of Coifman and Meyer [20] and Malvar [21], [22], such as the overlapping windows and the slope of the bell functions. Although PHLST can be defined quite generally as in [17], we restrict ourselves to the simplest case in

this paper. Let f be a function defined on the block domain $\overline{\Omega}$ as before. Then PHLST decomposes f into two components as $f = u + v$. The components u and v are referred to as the *polyharmonic component* and the *residual*, respectively. The polyharmonic component is obtained by solving *Laplace's equation*,

$$\Delta u = 0 \quad \text{in } \Omega, \tag{1}$$

with given boundary values

$$u = f \quad \text{on } \partial\Omega, \tag{2}$$

where $\Delta = \sum_{i=1}^{n} \partial^2/\partial x_i^2$ is the Laplace operator in $\mathbb{R}^n$. Note that Laplace's equation is the simplest case of the more general *polyharmonic equation*: $\Delta^m u = 0$, $m \in \mathbb{N}$. Eq. (2) is referred to as the *Dirichlet boundary condition* that enforces the function values of the solution u on the boundary $\partial\Omega$ to match those of the original signal f over there. Note that in 1D ($n = 1$), u is simply a straight line connecting two boundary points of an interval $\overline{\Omega}$. However, note also that in higher dimensions ($n \geq 2$), the solution of (1) with (2) is not a tensor product of algebraic polynomials in general. Now subtracting the u component from f gives us the residual $v = f - u$ satisfying $v = 0$ on $\partial\Omega$. Since the values of v on $\partial\Omega$ vanish, the extension of v by odd reflection with respect to the boundary $\partial\Omega$ is a C^1 periodic function on the extended domain if there is no intrinsic singularity in Ω. Therefore, one can apply Theorem 1 with $m = 1$, which implies that the Fourier sine coefficients of v decay rapidly, i.e., $O(\|\boldsymbol{k}\|^{-3})$. More precisely, we have the following theorem.

Theorem 2: Let Ω be a bounded rectangular domain in $\mathbb{R}^n$, and let $f \in C^2(\overline{\Omega})$, but non-periodic. Assume further that $\partial^3 f/\partial x_i^3$, $i = 1, \ldots, n$, exist and are of bounded variation. Furthermore, let $f = u + v$ be the PHLST representation, i.e., the polyharmonic component u is the solution of Laplace's equation (1) and the Dirichlet boundary condition (2), and $v = f - u$ is the residual. Then, the Fourier sine coefficient $b_{\boldsymbol{k}}$ of the residual v is of $O\left(\|\boldsymbol{k}\|^{-3}\right)$ for all $\boldsymbol{k} \neq \boldsymbol{0}$, where $\boldsymbol{k} = (k_1, \ldots, k_n) \in \mathbb{Z}_+^n$.

The proof of this theorem and its generalization can be found in [17]. The polyharmonic components can be computed quickly by utilizing the computationally-fast and numerically-accurate Laplace/Poisson solver developed by Averbuch, Braverman, Israeli, and Vozovoi [23], [24] as long as the boundary data are available. Combining this feature with the quickly decaying expansion coefficients of the residuals, the usefulness of PHLST to image approximation was demonstrated [16], [17]. However, there are two disadvantages in PHLST: 1) it has a more complicated data structure because the boundary pixels of each block must be stored separately from the inside of the block (although one can use the 1D version of PHLST to store the boundary pixels efficiently as shown in [17]); 2) it cannot utilize the infrastructure of the JPEG standard since it is based on DST instead of DCT.

III. Polyharmonic Local Cosine Transform

The disadvantages of PHLST described in the previous section motivated us to develop a cosine version of the transform, the *polyharmonic local cosine transform* (PHLCT). Our goal is to find a decomposition $f = u + v$ so that: 1) it is based on the Fourier cosine series expansion of the v components for the compatibility with the JPEG standard; and 2) the expansion coefficients decay as $O(\|\boldsymbol{k}\|^{-4})$ or faster. To do so, we want to make the residual v a C^2 periodic function after the even reflection at the block boundary so that we can apply Theorem 1 with $m = 2$. Thus, what we need to do is to match the normal derivatives of u across the block boundary with those of f because this forces the residual v to have the zero normal derivatives over there, which in turn allows the even reflection of v to have desired C^2 smoothness if there is no intrinsic singularity in the original data f in $\overline{\Omega}$. This can be explained more easily in 1D as follows. Let $\Omega = (0, 1)$ and $\widetilde{v}$ be an even-reflected version of v at the boundary $x = 0$ followed by its periodic extension of the interval $[-1, 1]$ into the entire real axis $\mathbb{R}$. Thus we have $\widetilde{v}(-x) = \widetilde{v}(x)$ for all $x \in \mathbb{R}$. It is clear $\widetilde{v}''(0)$ is well defined when x approaches to 0 from both sides. Also, the condition of the zero derivative at the boundary, $v'(0) = 0$ sets $\widetilde{v}'(0) = 0$. Similary, we can show that $\widetilde{v}''(1)$ is well defined and $\widetilde{v}'(1) = 0$. Hence, $\widetilde{v}$ becomes a C^2 periodic function with period 2.

Coming back to the higher dimensional case, therefore, we need to replace the Dirichlet boundary condition (2) by the *Neumann boundary condition*:

$$\begin{cases} \Delta u = 0 & \text{in } \Omega, \\ \dfrac{\partial u}{\partial \nu} = \dfrac{\partial f}{\partial \nu} & \text{on } \partial\Omega, \end{cases} \tag{3}$$

where the normal derivative $\partial f / \partial \nu$ at $\boldsymbol{x} \in \partial\Omega$ is defined as the directional derivative $\boldsymbol{\nu}(\boldsymbol{x}) \cdot \nabla f(\boldsymbol{x})$ along the normal vector $\boldsymbol{\nu}(\boldsymbol{x})$ that is perpendicular to the boundary $\partial\Omega$ at $\boldsymbol{x}$ pointing to the outside of Ω. However, Eq. (3) cannot have a solution in general, and we need to introduce the source term in the righthand side of Laplace's equation as follows:

$$\begin{cases} \Delta u = K & \text{in } \Omega, \\ \dfrac{\partial u}{\partial \nu} = \dfrac{\partial f}{\partial \nu} & \text{on } \partial\Omega, \end{cases} \tag{4}$$

where $K := \frac{1}{|\Omega|} \int_{\partial\Omega} \frac{\partial f}{\partial \nu} \, \mathrm{d}\sigma(\boldsymbol{x})$, $|\Omega|$ is the volume of the block Ω, and $\mathrm{d}\sigma(\boldsymbol{x})$ is a surface (or boundary) measure. Appendix I explains why we have to use (4), not (3). Laplace's equation with the source term is called *Poisson's equation* [25, Chap. 2]. It is a well-known fact that (4) has a unique solution modulo an additive constant; see e.g., [25, Sec. 2D].

The solution u to (4) is not just a function having the matching normal derivative at the boundary with the original function f; the following theorem further characterizes u:

Theorem 3: Let p be a $C^2(\overline{\Omega})$ function and $M(p)$ be the total squared curvature integral defined by

$$M(p) := \int_{\Omega} (\Delta p)^2 \, \mathrm{d}\boldsymbol{x} \, .$$

For any p with $\partial p/\partial\nu = \partial f/\partial\nu$ on $\partial\Omega$, the solution u to (4) satisfies $M(u) \leq M(p)$. In other words, the solution u is the minimizer of the total squared curvature integral on Ω.

The proof of this theorem can be found in Appendix II. Theorem 3 guarantees that the influence of the boundary data to the entire shape of u is kept to a minimum in terms of the mean curvature. The u component can also be viewed as a *smooth predictor* of the original block f solely based on the gradient information across the block boundary.

Once we get the solution u, the residual $v = f - u$ clearly satisfies $\partial v/\partial\nu = 0$ on $\partial\Omega$. Since $f \in C^2(\overline{\Omega})$ and Δu is constant in Ω, $\partial^2 u/\partial x_i \partial x_k$, $1 \leq i, k \leq n$, must be bounded as $\boldsymbol{x}$ approaches to $\partial\Omega$. Therefore, this leads to at least C^2 smoothness of the residual v across the block boundary when it is extended by the even reflection. Thus, we can apply Theorem 1 with $m = 2$, and consequently, we have the following.

Corollary 4: Let Ω be a bounded rectangular domain in $\mathbb{R}^n$, and let $f \in C^2(\overline{\Omega})$, but non-periodic. Assume further that $(\partial/\partial x_i)^3 f$, $i = 1, \ldots, n$, exist and are of bounded variation. Furthermore, let $f = u + v$ be the PHLCT representation, i.e., the polyharmonic component u is the solution to (4), and $v = f - u$ is the residual component. Then, the Fourier cosine coefficients of the residual v is of $O\left(\|\boldsymbol{k}\|^{-4}\right)$ for all $\boldsymbol{k} \neq \boldsymbol{0}$, where $\boldsymbol{k} = (k_1, \ldots, k_n) \in \mathbb{Z}_+^n$.

This fast decay of the Fourier cosine coefficients will shorten the code size when encoding the residual component. The function u can also be viewed as an interpolated image surface over the Ω, which can be contrasted with the other AC prediction methods such as QSFIT [1, Sec. 16.1] and LAKHANI [13]. We shall compare these methods in Section VII in details.

IV. COMPUTATION OF PHLCT FROM DCT COEFFICIENTS

In the previous section, we have derived the asymptotic behavior of PHLCT in the spatial-frequency domain in $\mathbb{R}^n$, $n \in \mathbb{N}$. From now on, we shall only deal with two-dimensional images (i.e., $n = 2$). Hence, we set $\Omega = (0, 1)^2 = \{(x, y) \,|\, 0 < x < 1, \, 0 < y < 1\}$, the unit square in $\mathbb{R}^2$. We shall now derive the DCT coefficients of the u component *directly* from the DCT coefficients of the original data f. Consequently, we shall achieve the PHLCT representation of the original data *in the DCT domain*, which leads us to our new compression-decompression algorithms as we shall examine in detail in Section VI.

Let $f(x, y)$ be in $C^2(\overline{\Omega})$. Assume that we are given the discretized version of f sampled at $(x_i, y_j) \in \Omega$ for $i, j = 0, 1, \ldots, N - 1$, where $x_i = (0.5 + i)/N$ and $y_j = (0.5 + j)/N$. Let $F = (F_{k_1, k_2})$ be a matrix of size $N \times N$, whose entries are the 2D-DCT coefficients of f defined as

$$F_{k_1, k_2} :=$$

$$\lambda_{k_2} \sqrt{\frac{2}{N}} \sum_{j=0}^{N-1} \left(\lambda_{k_1} \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} f(x_i, y_j) \cos \pi k_1 x_i \right) \cos \pi k_2 y_j, \tag{5}$$

where $k_1, k_2 = 0, 1, \ldots, N - 1$ and

$$\lambda_k := \begin{cases} 1/\sqrt{2} & \text{if } k = 0, \\ 1 & \text{otherwise.} \end{cases} \tag{6}$$

Let us denote the 2D-DCT coefficient matrix of the u component, i.e., the solution of (4), by U. The objective of this section is to compute U when the 2D-DCT coefficient matrix of the original data F is available. Clearly, once U is computed, the 2D-DCT coefficients of the residual are readily available via $V = F - U$.

A. Computing U assuming the ideal boundary condition

We shall now solve Poisson's equation with the Neumann boundary condition (4). Let us assume for the moment that the following discretized Neumann boundary data are available:

$$g_i^{(1)} := -f_y(x_i, 0), \quad g_i^{(2)} := f_y(x_i, 1), \quad g_j^{(3)} := -f_x(0, y_j),$$

$$g_j^{(4)} := f_x(1, y_j), \quad i, j = 0, 1, \ldots, N - 1, \tag{7}$$

where $f_x := \partial f(x, y)/\partial x$ and $f_y := \partial f(x, y)/\partial y$, respectively. In practice, the righthand sides of (7) are not readily available, and need to be estimated from the image samples. In Section IV-B, we shall derive a practical method to approximate the righthand sides of (7) using the DCT coefficients of the original data in the current and adjacent blocks, which will be used in our implementation. Let us now consider the following finite Fourier cosine series of the Neumann boundary data:

$$g^{(\ell)}(t) := \sqrt{\frac{2}{N}} \sum_{k=0}^{N-1} \lambda_k G_k^{(\ell)} \cos \pi k t,$$

$$= \sqrt{\frac{2}{N}} \left(\frac{G_0^{(\ell)}}{\sqrt{2}} + \sum_{k=1}^{N-1} G_k^{(\ell)} \cos \pi k t \right),$$

$$0 \leq t \leq 1, \quad \ell = 1, 2, 3, 4,$$

where λ_k was already defined in (6), and $\left\{G_k^{(\ell)}\right\}_{0 \le k \le N-1}$ are the 1D-DCT coefficients determined by the boundary data $\left\{g_j^{(\ell)}\right\}_{0 \le j \le N-1}$ in (7). Clearly $g^{(\ell)}(t_j) = g_j^{(\ell)}$, where $t_j = (0.5+j)/N$, $j = 0, 1, \ldots, N-1$, and $\ell = 1, \ldots, 4$. Now, it is straightforward to show that for $\ell = 1$, the function

$$u^{(1)}(x,y) := \sqrt{\frac{2}{N}} \left(\frac{G_0^{(1)}}{\sqrt{2}} \frac{(y-1)^2}{2} \right.$$
$$\left. + \sum_{k=1}^{N-1} G_k^{(1)} \frac{\cosh \pi k(y-1)}{\pi k \sinh \pi k} \cos \pi k x \right)$$

satisfies the following Poisson's equation with the Neumann boundary condition:

$$\begin{cases} \Delta u^{(1)} = \dfrac{1}{|\Omega|} \displaystyle\int_0^1 g^{(1)}(t)\,\mathrm{d}t = \dfrac{G_0^{(1)}}{\sqrt{N}} & \text{in } \Omega, \\[2ex] \dfrac{\partial u^{(1)}}{\partial \nu} = \begin{cases} g^{(1)} & \text{on} \quad \Gamma^{(1)}, \\ 0 & \text{on} \quad \partial\Omega \backslash \Gamma^{(1)}, \end{cases} \end{cases}$$

where $\Gamma^{(1)} := \{(x,y) \in \partial\Omega \,|\, 0 \le x \le 1,\, y = 0\}$. See also Figure 1. Similarly, we can derive $u^{(\ell)}$ for $\ell = 2, 3, 4$, and consequently we obtain the solution of (4) as follows:

$$u(x,y) = u^{(1)}(x,y) + u^{(2)}(x,y) + u^{(3)}(x,y) + u^{(4)}(x,y)$$
$$= \sqrt{\frac{2}{N}} \sum_{k=0}^{N-1} \lambda_k \left\{ \left(G_k^{(1)} \psi_k(y-1) + G_k^{(2)} \psi_k(y) \right) \cos \pi k x \right.$$
$$\left. + \left(G_k^{(3)} \psi_k(x-1) + G_k^{(4)} \psi_k(x) \right) \cos \pi k y \right\} + c, \tag{8}$$

where c is an arbitrary constant that will be determined shortly and

$$\psi_k(t) := \begin{cases} t^2/2 & \text{if } k = 0, \\ \cosh \pi k t/(\pi k \sinh \pi k) & \text{otherwise.} \end{cases}$$

We note that we derived the solution (8) by modifying the highly accurate Dirichlet problem solver developed by Averbuch, Israeli, and Vozovoi [23] for our Neumann problem.

Finally, by applying 2D DCT to (8), we obtain U_{k_1,k_2}, $k_1, k_2 = 0, 1, \ldots, N-1$:

$$U_{k_1,k_2} = G_{k_1}^{(1)} \eta_{k_1,k_2} + G_{k_1}^{(2)} \eta_{k_1,k_2}^* + G_{k_2}^{(3)} \eta_{k_2,k_1} + G_{k_2}^{(4)} \eta_{k_2,k_1}^*, \tag{9}$$

where

$$\eta_{k,m} := \lambda_m \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \psi_k(x_i - 1) \cos \pi m x_i \,,$$

$$\eta_{k,m}^* := \lambda_m \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \psi_k(x_i) \cos \pi m x_i \,. \tag{10}$$

TABLE I

NUMERICAL VALUES OF $\eta_{k,m}$ COMPUTED UP TO FOUR DECIMAL PLACES

	$m=0$	$m=1$	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$	$m=7$
$k=0$	0.6641	0.4026	0.0986	0.0421	0.0221	0.0126	0.0070	0.0032
$k=1$	0.4027	0.2000	0.0783	0.0376	0.0206	0.0120	0.0067	0.0031
$k=2$	0.0988	0.0785	0.0480	0.0283	0.0171	0.0104	0.0060	0.0028
$k=3$	0.0425	0.0380	0.0285	0.0197	0.0132	0.0085	0.0051	0.0024
$k=4$	0.0229	0.0214	0.0177	0.0135	0.0097	0.0066	0.0041	0.0020
$k=5$	0.0139	0.0132	0.0115	0.0093	0.0071	0.0051	0.0032	0.0016
$k=6$	0.0090	0.0087	0.0078	0.0066	0.0052	0.0038	0.0025	0.0012
$k=7$	0.0061	0.0060	0.0054	0.0047	0.0038	0.0028	0.0019	0.0009

Note that $\eta_{k,m}$ and $\eta_{k,m}^*$ are independent from any image sample values: once the number of samples N is fixed, they are completely determined and precomputed independently from an input image. For the convenience of the reader, the values of $\eta_{k,m}$ for $N=8$ are listed in Table I. Finally, we determine the constant c in (8) so that $U_{0,0}=0$. In fact, from (9), (10), we can explicitly determine this constant:

$$c = -\frac{4N^2-1}{24N^{2.5}}\left(G_0^{(1)} + G_0^{(2)} + G_0^{(3)} + G_0^{(4)}\right).$$

Thanks to Corollary 4, the decay rate of the DCT coefficients of the residual $V_{\boldsymbol{k}} = F_{\boldsymbol{k}} - U_{\boldsymbol{k}}$ for $\boldsymbol{k} = (k_1, k_2) \neq (0,0)$ should be $O(\|\boldsymbol{k}\|^{-4})$.

B. Approximation of the Neumann boundary condition

In practice, the Neumann boundary data, i.e., the righthand sides of (7), need to be estimated from the image samples. To do so, we need not only the image samples of the current block but also those of the adjacent blocks. Let us consider the four blocks adjacent to Ω as shown in Figure 1. We assume that f is also defined on these adjacent blocks and sampled with the same rate as the current (or center) block Ω. Let us denote the image samples on each block $\Omega^{(s,t)}$ in Fig. 1 by $f_{i,j}^{(s,t)} := f(x_i + s, y_j + t)$, $i, j = 0, 1, \ldots, N-1$, where $s, t \in \{-1, 0, 1\}$. For convenience, let us also set $\Omega^{(0,0)} = \Omega$ and $f_{i,j}^{(0,0)} =$
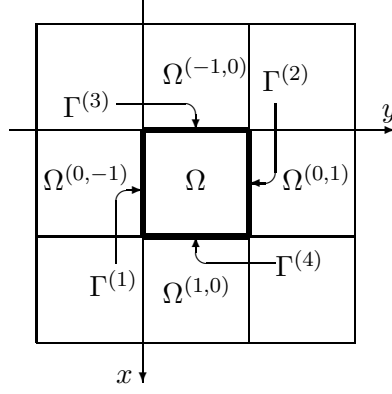
Fig. 1.   The configuration of the adjacent blocks and boundary segments of Ω.

$f(x_i, y_j)$. We now approximate the normal derivatives at the boundary of Ω as follows.

$$g_i^{(1)} \simeq X_i^{(-1)} - X_i^{(0)}, \quad g_i^{(2)} \simeq X_i^{(1)} - X_i^{(0)},$$

$$g_j^{(3)} \simeq Y_j^{(-1)} - Y_j^{(0)}, \quad g_j^{(4)} \simeq Y_j^{(1)} - Y_j^{(0)}, \tag{11}$$

where

$$X_i^{(t)} := \frac{1}{N} \sum_{j=0}^{N-1} f_{i,j}^{(0,t)}, \ Y_j^{(s)} := \frac{1}{N} \sum_{i=0}^{N-1} f_{i,j}^{(s,0)},$$

are the row-wise and column-wise averages, respectively. Let $F^{(s,t)} = \left(F_{k_1,k_2}^{(s,t)} \right)$ be the DCT coefficient matrix of $f^{(s,t)}$ on $\Omega^{(s,t)}$. Then, using (5) and (11), we can express each 1D-DCT coefficient $G_k^{(1)}$ of the Neumann data $g^{(1)}$ as a simple function of *the first column* of the 2D-DCT coefficients of the current and the adjacent blocks as follows:

$$G_k^{(1)} = \lambda_k \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} g_i^{(1)} \cos \pi k x_i$$

$$\simeq \lambda_k \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \left\{ X_i^{(-1)} - X_i^{(0)} \right\} \cos \pi k x_i$$

$$= \frac{1}{\sqrt{N}} (F_{k,0}^{(0,-1)} - F_{k,0}), \quad k = 0, 1, \ldots, N-1,$$

Similarly we have

$$G_k^{(2)} \simeq \frac{1}{\sqrt{N}} (F_{k,0}^{(0,1)} - F_{k,0}), G_k^{(3)} \simeq \frac{1}{\sqrt{N}} (F_{0,k}^{(-1,0)} - F_{0,k}),$$

$$G_k^{(4)} \simeq \frac{1}{\sqrt{N}} (F_{0,k}^{(1,0)} - F_{0,k}),$$

which are simple functions of either the first column or the first row of the 2D-DCT coefficients of the current and adjacent blocks. Inserting these into (9), we have the U_{k_1,k_2} for the approximate discretized Neumann data as

$$
\begin{aligned}
U_{k_1,k_2} = \frac{1}{\sqrt{N}} \Big\{ & (F_{k_1,0}^{(0,-1)} - F_{k_1,0})\eta_{k_1,k_2} \\
& + (F_{k_1,0}^{(0,1)} - F_{k_1,0})\eta_{k_1,k_2}^* + (F_{0,k_2}^{(-1,0)} - F_{0,k_2})\eta_{k_2,k_1} \\
& + (F_{0,k_2}^{(1,0)} - F_{0,k_2})\eta_{k_2,k_1}^* \Big\} ,
\end{aligned}
\tag{12}
$$

for all $k_1,\ k_2 = 0, 1, \dots, N-1$ except $k_1 = k_2 = 0$. Strictly speaking, Eq. (12) should be an approximation rather than an equality. However, we assume U_{k_1,k_2} as the righthand side of Eq. (12) from now on because we only use the approximate Neumann boundary data Eq. (11) in practice. One can view Eq. (12) as the new definition of U_{k_1,k_2}. As we discussed at the end of Section IV-A, $U_{0,0}$, the DC component of u, vanishes. Hence the DC component of the original data f is carried over to that of the v component, i.e., $V_{0,0} = F_{0,0}$. This will become important for our further modification of PHLCT algorithm in Section V and our new compression-decompression algorithms in Section VI.

In the above discussion, we have approximated the first normal derivatives at the block boundary by using the DCT coefficients of the current and its adjacent blocks. Clearly, this approximation scheme is not applicable to the outmost blocks of an input image where some of their adjacent blocks do not exist. In such a case, we set the Neumann boundary function simply by zero, i.e., $g^{(\ell)}(t) = 0$ for appropriate ℓ.

V. MODIFYING PHLCT FOR PRACTICE

At this point, let us consider what information must be stored in the forward PHLCT described in the previous section in order to recover the original data f on the block Ω. Both U and V are of course necessary in order to recover f exactly. However, we do not need to store all these $2N^2$ coefficients in PHLCT. Note that each U_{k_1,k_2} is a simple function of the first column or row of $F^{(s,t)}$ with $(s,t) \in \mathcal{I}_5 :=$ $\{(0,-1), (-1,0), (0,0), (1,0), (0,1)\}$. Therefore, in order to exactly recover f of the current block, we just need $N^2 + 6N - 2$ coefficients: N^2 for all the entries of V; $6N$ for $F_{k_1,0}$, F_{0,k_2}, $F_{k_1,0}^{(0,-1)}$, $F_{k_1,0}^{(0,1)}$, $F_{0,k_2}^{(-1,0)}$, $F_{0,k_2}^{(1,0)}$ where $k_1, k_2 = 0, 1, \dots N-1$; and -2 for counting the DC component $F_{0,0}$ three times (note that $F_{0,0} = V_{0,0}$). Considering an application of PHLCT to image compression, this situation is not desirable although it may be acceptable for other purposes such as interpolation, zooming, feature extraction, etc. What we want to have is to recover f exactly from V with the smallest possible number of DCT coefficients from the adjacent blocks. It turns out that we can manage to recover f exactly by

$N^2 + 4$ coefficients: N^2 for V and 4 for the DC components of the adjacent blocks. To do so, we need to modify both u and v components in the original PHLCT. In the forward PHLCT, we construct the modified version of U denoted by $\widetilde{U}$ as follows:

$$\widetilde{U}_{k_1,k_2} := \begin{cases} 0 & \text{if } k_1 = k_2 = 0; \\ \frac{1}{\sqrt{N}}\left\{(F_{0,0}^{(-1,0)} - F_{0,0})\eta_{0,k_1} \right. \\ \left. +(F_{0,0}^{(1,0)} - F_{0,0})\eta_{0,k_1}^*\right\} & \text{if } k_1 \neq 0 = k_2; \\ \frac{1}{\sqrt{N}}\left\{(F_{0,0}^{(0,-1)} - F_{0,0})\eta_{0,k_2} \right. \\ \left. +(F_{0,0}^{(0,1)} - F_{0,0})\eta_{0,k_2}^*\right\} & \text{if } k_1 = 0 \neq k_2; \\ U_{k_1,k_2} & \text{otherwise,} \end{cases} \tag{13}$$

where U_{k_1,k_2} is defined in (12). In essence, the only difference between U and $\widetilde{U}$ are their first columns and rows, and those of $\widetilde{U}$ can be computed *only using the DC components of the current and adjacent blocks of the originals*. On the other hand, the first column and row of F are necessary to compute those of U; examine (12) by setting $k_1 = 0$ or $k_2 = 0$. Once we obtain $\widetilde{U}$, then we complete the forward PHLCT by computing the modified residual $\widetilde{V}$ simply by $F - \widetilde{U}$. More explicitly, we compute

$$\widetilde{V}_{k_1,k_2} :=$$
$$\begin{cases} F_{0,0} & \text{if } k_1 = k_2 = 0; \\ F_{k_1,0} - \frac{1}{\sqrt{N}}\left\{(F_{0,0}^{(-1,0)} - F_{0,0})\eta_{0,k_1} \right. \\ \left. +(F_{0,0}^{(1,0)} - F_{0,0})\eta_{0,k_1}^*\right\} & \text{if } k_1 \neq 0 = k_2; \\ F_{0,k_2} - \frac{1}{\sqrt{N}}\left\{(F_{0,0}^{(0,-1)} - F_{0,0})\eta_{0,k_2} \right. \\ \left. +(F_{0,0}^{(0,1)} - F_{0,0})\eta_{0,k_2}^*\right\} & \text{if } k_1 = 0 \neq k_2; \\ F_{k_1,k_2} - U_{k_1,k_2} & \text{otherwise.} \end{cases} \tag{14}$$

As for the storage, we only need to store $\widetilde{V}$, not $\widetilde{U}$ at all.

Let us now consider the inverse PHLCT. Given the 2D-DCT coefficients $\widetilde{V}$, it is clear that we can reconstruct the original data f as follows: 1) Recover the first column and row of $\widetilde{U}$ using the DC components, $F_{0,0}^{(s,t)}(= \widetilde{V}_{0,0}^{(s,t)})$, $(s,t) \in \mathcal{I}_5$ via (13); 2) Recover the first column and row of $F^{(s,t)}$, $(s,t) \in \mathcal{I}_5$ by summing those of $\widetilde{U}$ and $\widetilde{V}$ (see (13) and (14)); 3) Recover other entries of $\widetilde{U}$ via (13), (12) and the results of Step 2; 4) Set $F = \widetilde{U} + \widetilde{V}$; 5) Apply Inverse 2D DCT to F to recover f.

It is now obvious that the forward and inverse PHLCT algorithms of this modified version, in particular, the inverse algorithm, become much simpler than those of the original PHLCT in the previous section.

However, it is natural to ask what we have lost instead. Because of the uniqueness theorem [25, Sec. 2D, 3E] (modulo an additive constant), it is true that, the $\widetilde{u}$ component, the inverse DCT of $\widetilde{U}$, does not satisfy Poisson's equation (4) unlike the u component in the previous section. Strictly speaking, this means that $\widetilde{u}$ does not satisfy Theorem 3 anymore. However, in terms of the Neumann boundary condition, it is easy to show that $\partial \widetilde{u}/\partial \nu = \partial u/\partial \nu$ on $\partial \Omega$, since the difference

$$
\begin{aligned}
u - \widetilde{u} = \frac{\sqrt{2}}{N\sqrt{N}} \sum_{k=1}^{N-1} \Big[& \{ (F_{k,0}^{(0,-1)} - F_{k,0})\eta_{k,0} \\
& + (F_{k,0}^{(0,1)} - F_{k,0})\eta_{k,0}^* \} \cos \pi k x \\
& + \{ (F_{0,k}^{(-1,0)} - F_{0,k})\eta_{k,0} + (F_{0,k}^{(1,0)} - F_{0,k})\eta_{k,0}^* \} \cos \pi k y \Big]
\end{aligned}
$$

clearly satisfies $\partial(u - \widetilde{u})/\partial \nu = 0$ on $\partial \Omega$ thanks to the cosine terms above. Thus, it is still true that $\partial \widetilde{v}/\partial \nu = 0$ on $\partial \Omega$, and consequently $\widetilde{V}_{\boldsymbol{k}} = O(\|\boldsymbol{k}\|^{-4})$. Hence in practice, we have not lost anything.

From now on, we shall not use the original PHLCT of the previous section, and only use the modified version developed in this section. Therefore, to simplify our notation, we shall drop $\widetilde{}$ and simply use U and V for $\widetilde{U}$ and $\widetilde{V}$.

VI. APPLICATION OF PHLCT TO IMAGE COMPRESSION

In this section we propose new compression-decompression algorithms using PHLCT and show its practical advantage over the JPEG Baseline method.

A. Brief review of the lossy JPEG image compression standard

In the JPEG Baseline method, the amount of compression and the quality of reconstructed images is controlled by an 8×8 quantization table. Let Q be a quantization table to be used and let $Q_{\boldsymbol{k}} (\geq 1)$ be the $\boldsymbol{k}$th entry of Q where $\boldsymbol{k} \in \mathcal{K} := \{(k_1, k_2) \mid k_1, k_2 = 0, 1, 2, \ldots, 7 \}$. Then DCT coefficients $\{F_{\boldsymbol{k}}\}$ in each block is quantized simply by

$$
i_{\boldsymbol{k}} := \text{round}(F_{\boldsymbol{k}}/Q_{\boldsymbol{k}}); \quad F_{\boldsymbol{k}}^Q := Q_{\boldsymbol{k}} \times i_{\boldsymbol{k}}, \quad \boldsymbol{k} \in \mathcal{K}. \tag{15}
$$

This scheme truncates any DCT coefficient $F_{\boldsymbol{k}}$ with $|F_{\boldsymbol{k}}| < Q_{\boldsymbol{k}}/2$. Although the JPEG standard allows one to use any user-supplied quantization table, the most commonly used one is a scalar multiple of the standard Luminance Quantization Table (LQT) shown in Table II. These values were determined by extensive psychophysical experiments on the visibility of the DCT basis vectors [1, Sec. 4.1.3].

The user can change the values of this standard LQT by specifying a parameter called the *quality factor* ranging from 0 (the worst reconstruction quality and the best amount of compression) to 100 (the

TABLE II

THE LUMINANCE QUANTIZATION TABLE RECOMMENDED BY THE JPEG STANDARD.

	$k_2 = 0$	$k_2 = 1$	$k_2 = 2$	$k_2 = 3$	$k_2 = 4$	$k_2 = 5$	$k_2 = 6$	$k_2 = 7$
$k_1 = 0$	16	11	10	16	24	40	51	61
$k_1 = 1$	12	12	14	19	26	58	60	55
$k_1 = 2$	14	13	16	24	40	57	69	56
$k_1 = 3$	14	17	22	29	51	87	80	62
$k_1 = 4$	18	22	37	56	68	109	103	77
$k_1 = 5$	24	35	55	64	81	104	113	92
$k_1 = 6$	49	64	78	87	103	121	120	101
$k_1 = 7$	72	92	95	98	112	100	103	99

best reconstruction quality and the worst amount of compression). Let T be an 8×8 matrix whose entries are the standard LQT shown in Table II. Let q be the qualify factor. Then, the actual quantization table Q for 8 bit grayscale images is typically computed as

$$
\begin{aligned}
Q_{\boldsymbol{k}} &= Q_{\boldsymbol{k}}(T_{\boldsymbol{k}}; q) \\
&= \begin{cases}
\min\left(\text{round}\left(\frac{50 \cdot T_k}{q}\right), 255\right) & \text{if } 0 \leq q \leq 50; \\
\max\left(\text{round}\left(\frac{(100-q) \cdot T_k}{50}\right), 1\right) & \text{if } 50 \leq q \leq 100.
\end{cases}
\end{aligned}
\tag{16}
$$

Note that when $q = 100$, then $Q_{\boldsymbol{k}} = 1$ for all $\boldsymbol{k}$. In this case the loss due to the quantization is minimum (only the rounding operation is applied directly to the DCT coefficients).

After the quantization, the DC component is encoded as the difference from the DC term of the previous block because the DC components of the adjacent blocks usually correlates strongly and the differential encoding further reduces such correlation or redundancy. The integer sequence $\{i_{\boldsymbol{k}}\}$ in (15) is then ordered into the "zig-zag" sequence, and encoded by some lossless entropy coder, e.g., the Huffman or arithmetic coder, to further compress the sequence before transmitting or storing it.

To decompress the compressed representation, the integer sequence $\{i_{\boldsymbol{k}}\}$ in (15) is first recovered by the decoding algorithm corresponding to the lossless encoder followed by undoing the zig-zag ordering. Then the quantized coefficient matrix F^Q is recovered via (15). Finally, the 2D inverse DCT (IDCT) is applied to F^Q. For the details, see [1], [2].

B. "Full mode" PHLCT: Further reduction of JPEG-DCT bit-rate

As shown in Section III, PHLCT can improve the decay rate of the DCT coefficients up to $O(\|\boldsymbol{k}\|^{-4})$ compared to $O(\|\boldsymbol{k}\|^{-2})$ if the approximations of the normal derivatives given by (11) are effective. This means that *for the same prescribed quality, the bit-rate should be reduced by PHLCT.* We shall demonstrate our claim by numerical experiments in Section VII.

Let us now propose our new compression/decompression algorithm using our modified PHLCT detailed in Section V. We shall call our new algorithm the "full mode" PHLCT (or simply PHLCT for short) because this is a full and direct application of our decomposition and reconstruction algorithms in Section V to image compression/decompression, which requires modifying both the encoder part and the decoder part of the JPEG Baseline method. We can summarize our proposed algorithm as follows:

Compression:

- Divide an input image into blocks of 8×8 pixels.

- Apply DCT to compute F in each block.

○ Compute U via (13) and (12) and set $V = F - U$.

- Quantize V via $i_{\boldsymbol{k}} = \text{round}(V_{\boldsymbol{k}}/Q_{\boldsymbol{k}})$

  and encode $\{i_{\boldsymbol{k}}\}$ in the same way as JPEG.

Reconstruction:

- Decode $\{i_{\boldsymbol{k}}\}$ and reconstruct V^Q via $V_{\boldsymbol{k}}^Q = Q_{\boldsymbol{k}} \times i_{\boldsymbol{k}}$.

○ Compute U^Q via (14), (13), and (12) using V^Q,

  and construct $U^Q + V^Q$.

- Apply IDCT to $U^Q + V^Q$ in each block.

The steps marked by white dots are added to the original JPEG standard, and the other steps are exactly the same as the JPEG standard. The new step added in the reconstruction procedure is exactly the same as the inverse PHLCT algorithm described in the previous section except that the available coefficients are quantized. The increment of the computational cost compared to the original JPEG standard is mainly due to the computation of U in the encoder and U^Q in the decoder. It is clear from (12) and (13) that the computational cost of U or U^Q is approximately $3\,C_1 N^2 + 4\,C_2 N^2$ for each block of size $N \times N$ pixels, where C_1 and C_2 denote the unit costs for the arithmetic addition and multiplication, respectively. The values of η_{k_1,k_2}, and η_{k_1,k_2}^* are common in all blocks and computed in $O(N^2 \log_2 N)$ operations

via FFT; see Table I. These quantities should be computed only once in the encoder and in the decoder. In addition, both the encoder and the decoder clearly require $C_1 N^2$ operations per block for computing $F - U$ and $U^Q + V^Q$, respectively.

C. "Partial mode" of PHLCT: Restoration of truncated JPEG-DCT coefficients

It is of practical interest to have a new decompression algorithm that can: 1) reproduce images of improved quality compared to the ones obtained by the JPEG Baseline method while keeping the same bit rate; and 2) accept JPEG files, i.e., image data already compressed by the JPEG standard. In this section, we propose an algorithm to do just that. This means that our algorithm does not touch the encoder and assumes the availability of the quantized DCT coefficients $F_{\boldsymbol{k}}^Q$ in each block instead of $V_{\boldsymbol{k}}^Q$. This new algorithm will be referred to as the *partial mode* PHLCT (or PPHLCT for short) because it only modifies the decorder part of the JPEG Baseline method unlike the full mode PHLCT.

In short, our key idea in PPHLCT is the use of $U_{\boldsymbol{k}}^Q$ to "fill in" the truncated $F_{\boldsymbol{k}}^Q$ by the encoder. Let us now discuss this idea in details. The amount of data compression in the lossy JPEG standard mainly comes from the truncation of the high frequency DCT coefficients. Now, the asymptotic behavior of $F_{\boldsymbol{k}}$ and the residual $V_{\boldsymbol{k}}$ are given by

$$F_{\boldsymbol{k}} = O(\|\boldsymbol{k}\|^{-2}),\ V_{\boldsymbol{k}} = O(\|\boldsymbol{k}\|^{-4})\ \text{ as }\ \|\boldsymbol{k}\| \to \infty\,.$$

Hence, we have $F_{\boldsymbol{k}} \simeq U_{\boldsymbol{k}}$ as $\|\boldsymbol{k}\| \to \infty$, although we can only reach $\|\boldsymbol{k}\| = 7\sqrt{2} \approx 10$ when each block consists of 8×8 pixels. However, this observation suggests a possibility to approximately restore the truncated higher frequency components using $U_{\boldsymbol{k}}$. It is important to note that $U_{\boldsymbol{k}}$ can be well approximated even after the quantization thanks to (12) and (13) in which $F_{\boldsymbol{k}}^{(s,t)}$ is replaced by $F_{\boldsymbol{k}}^{(s,t)Q}$. This is because those low frequency coefficients used in the righthand side of (12) and (13) are less affected by the quantization process compared to the higher frequency coefficients. Since it is obvious from (10) that $|\eta_{k_1,k_2}| = |\eta_{k_1,k_2}^*|$, we have the following error estimate using (12) and (13):

$$\begin{aligned}|U_{\boldsymbol{k}} - U_{\boldsymbol{k}}^Q| &\leq \frac{1}{\sqrt{N}}(2E_{k_1,0}|\eta_{k_1,k_2}| + 2E_{k_1,0}|\eta_{k_1,k_2}^*| \\ &\quad + 2E_{0,k_2}|\eta_{k_2,k_1}| + 2E_{0,k_2}|\eta_{k_2,k_1}^*|) \\ &= \frac{4}{\sqrt{N}}(E_{k_1,0}|\eta_{k_1,k_2}| + E_{0,k_2}|\eta_{k_2,k_1}|)\end{aligned}$$

for all $\boldsymbol{k} = (k_1, k_2)$, where $E_{\boldsymbol{k}}$ denotes an estimation of the quantization error $\max\limits_{(s,t)\in\mathcal{I}_5} |F_{\boldsymbol{k}}^{(s,t)} - F_{\boldsymbol{k}}^{(s,t)\,Q}|$.

From these considerations, we propose the following scheme for the approximate restoration of the truncated DCT coefficients:

$$\text{Replace } F_{\boldsymbol{k}}^Q \text{ by } F_{\boldsymbol{k}}^Q + d_{\boldsymbol{k}}, \ \ d_{\boldsymbol{k}} := \begin{cases} U_{\boldsymbol{k}}^Q \text{ if } \boldsymbol{k} \in \mathcal{K}_t, \\ \\ 0 \ \ \text{ if } \boldsymbol{k} \in \mathcal{K} \backslash \mathcal{K}_t, \end{cases} \tag{17}$$

where $\mathcal{K}_t := \left\{ \boldsymbol{k} \in \mathcal{K} \ \middle| \ F_{\boldsymbol{k}}^Q = 0 \text{ and } |U_{\boldsymbol{k}}^Q| < Q_{\boldsymbol{k}}/2 \right\}$. In words, the truncated DCT coefficients $F_{\boldsymbol{k}}^Q \, (= 0)$ is replaced by $U_{\boldsymbol{k}}^Q \, (\neq 0)$ if $|U_{\boldsymbol{k}}^Q|$ is as small as $|F_{\boldsymbol{k}}|$. Note also that $d_{0,0} = 0$ by definition.

Unfortunately, this method is not effective enough to eliminate the blocking artifacts that become noticeable and annoying particularly in the case of low bit-rate compression. We therefore propose an additional procedure to reduce the blocking artifacts. Let us now consider the reduction of the blocking artifacts on the boundary $\Gamma^{(1)}$ between the current block Ω and the neighboring block $\Omega^{(0,-1)}$ (see also Fig. 1). The reconstructed image $f^Q(x, y)$ obtained from F^Q of (17) still has the following *mean* discontinuity across the boundary $\Gamma^{(1)}$:

$$\delta^{(1)} = \frac{1}{N} \sum_{i=0}^{N-1} \left(f^Q(x_i, 0-) - f^Q(x_i, 0+) \right)$$

$$= \frac{\sqrt{2}}{N} \sum_{k=0}^{N-1} \lambda_k (F_{0,k}^{(0,-1)Q} \cos \pi k - F_{0,k}^Q),$$

where $f^Q(x_i, 0-)$ and $f^Q(x_i, 0+)$ denote the left and right limit values at $y = 0$, respectively, and the second equality follows from the definition of the DCT coefficients (5). In order to remove this discontinuity, we add the following quadratic polynomials to $f^Q(x, y)$ on $\Omega^{(0,-1)}$ and Ω, respectively:

$$\phi^{(0,-1)}(x, y) := (\alpha y - 1)(y + 1) \frac{\delta^{(1)}}{2},$$

$$\phi(x, y) := (\alpha y - 1)(y - 1) \frac{\delta^{(1)}}{2},$$

where α denotes a constant to be determined. Since $\phi^{(0,-1)}(x_i, 0) = -\delta^{(1)}/2$ and $\phi(x_i, 0) = \delta^{(1)}/2$, the mean discontinuity across $\Gamma^{(1)}$ vanishes as follows:

$$\frac{1}{N} \sum_{i=0}^{N-1} \{ (f^Q(x_i, 0-) + \phi^{(0,-1)}(x_i, 0))$$

$$- (f^Q(x_i, 0+) + \phi(x_i, 0)) \} = 0 \, .$$

These polynomials do not affect the mean discontinuity computation across the boundary segments other than $\Gamma^{(1)}$ because: 1) $\phi^{(0,-1)}(x, -1) = 0$ and $\phi(x, 1) = 0$; and 2) the mean values of $\phi^{(0,-1)}(x, y)$ and $\phi(x, y)$ along the other boundary segments of Ω and $\Omega^{(0,-1)}$ also vanish if we set the constant

$\alpha = (6N^2)/(2N^2 + 1)$, i.e.,

$$\sum_{j=0}^{N-1} \phi^{(0,-1)}(0, y_j) = \sum_{j=0}^{N-1} \phi^{(0,-1)}(1, y_j) = 0\,,$$

$$\sum_{j=0}^{N-1} \phi(0, y_j) = \sum_{j=0}^{N-1} \phi(1, y_j) = 0\,.$$

Applying this procedure to $\Gamma^{(2)}$, $\Gamma^{(3)}$, and $\Gamma^{(4)}$, we can remove the discontinuities at the block boundary of Ω. In fact, we add the DCT coefficient matrix $P = (P_{k_1, k_2})$ of the following quadratic polynomial to $F^Q(x, y)$ on each block:

$$p(x, y) := (\alpha y - 1)(y - 1)\frac{\delta^{(1)}}{2} - (\alpha(1 - y) - 1)y\frac{\delta^{(2)}}{2}$$
$$+ (\alpha x - 1)(x - 1)\frac{\delta^{(3)}}{2} - (\alpha(1 - x) - 1)x\frac{\delta^{(4)}}{2}, \tag{18}$$

where

$$\delta^{(2)} = \frac{\sqrt{2}}{N} \sum_{k=0}^{N-1} \lambda_k (F_{0,k}^{(0,1)Q} - F_{0,k}^Q \cos \pi k)\,,$$

$$\delta^{(3)} = \frac{\sqrt{2}}{N} \sum_{k=0}^{N-1} \lambda_k (F_{k,0}^{(-1,0)Q} \cos \pi k - F_{k,0}^Q)\,,$$

$$\delta^{(4)} = \frac{\sqrt{2}}{N} \sum_{k=0}^{N-1} \lambda_k (F_{k,0}^{(1,0)Q} - F_{k,0}^Q \cos \pi k)\,.$$

By the straightforward computation using (5) and (18), we have

$$P_{k_1, k_2} = \begin{cases} 0 & \text{if } k_1 = k_2 = 0; \\ \sqrt{N}\left(\gamma_{k_1}\delta^{(3)} - \gamma_{k_1}^*\delta^{(4)}\right) & \text{if } k_1 \neq 0 = k_2; \\ \sqrt{N}\left(\gamma_{k_2}\delta^{(1)} - \gamma_{k_2}^*\delta^{(2)}\right) & \text{if } k_1 = 0 \neq k_2; \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

where

$$\gamma_k := \lambda_k \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \{(\alpha x_i - 1)(x_i - 1)\} \cos \pi k x_i\,,$$

$$\gamma_k^* := \lambda_k \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \{(\alpha(1 - x_i) - 1)x_i\} \cos \pi k x_i\,.$$

TABLE III

NUMERICAL VALUES OF γ_k, γ_k^* COMPUTED UP TO FOUR DECIMAL PLACES

	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$
γ_k	0.0000	0.8053	0.5869	0.0842	0.1316	0.0251	0.0417	0.0063
γ_k^*	0.0000	0.8053	-0.5869	0.0842	-0.1316	0.0251	-0.0417	0.0063

Based on the above discussions, we propose the following PPHLCT algorithm that provides better approximation to the original DCT coefficient F than the JPEG Baseline method does:

$\left\{ \right.$ **Compression:** Same as the JPEG standard.

$\left\{ \right.$
Reconstruction:

• Decode the compressed sequence and reconstruct F^Q via (15).

∘ Compute U^Q via (13), (12).

∘ Compute $d_{\boldsymbol{k}}$ as (17).

∘ Compute $P_{\boldsymbol{k}}$ via (19).

∘ Replace $F_{\boldsymbol{k}}^Q$ by $F_{\boldsymbol{k}}^Q + d_{\boldsymbol{k}} + P_{\boldsymbol{k}}$ if $|d_{\boldsymbol{k}} + P_{\boldsymbol{k}}| \leq Q_{\boldsymbol{k}}/2$.

• Apply IDCT to F^Q in each block.

The computational cost of PPHLCT is slightly larger than that of the PHLCT decoder due to the computation of $d_{\boldsymbol{k}}$ and $P_{\boldsymbol{k}}$. The computation of $d_{\boldsymbol{k}}$ requires $3N^2$ comparisons per block. if we implement the absolute value computation in (17) simply by the comparison operation. The cost for $P_{\boldsymbol{k}}$ is essentially negligible; in fact, it is of $O(N)$ per block instead of $O(N^2)$ since $P_{\boldsymbol{k}}$ has a nonzero value only if $k_1 \cdot k_2 = 0$ and the values of γ_k, γ_k^* are common for all the blocks.

For the convenience of the reader, we list the numerical values of γ_k, γ_k^* in Table III.

VII. NUMERICAL EXPERIMENTS

In this section, we describe our numerical experiments and demonstrate the effectiveness of our methods. The grayscale images "Gabor", "Lenna", and "Barbara" shown in Figure 2 (512×512 pixels,

(a) Gabor   (b) Lenna   (c) Barbara

Fig. 2.   Original images (512×512 pixels, 8 bits/pixel grayscale).

8 bits/pixel) are used in the experiments. The image "Gabor" is a synthetic image, which is a two-dimensional Gabor function sampled on the regular grid.

A. How close U and U^Q are to F ?

As we mentioned in Section III, the u component (or its DCT representation U) can be viewed as a predictor of f (or its DCT representation F) using the gradient information across the block boundary. Clearly, the U component with good prediction capability helps encoding the residual V in PHLCT. It is also of our interest to know how well the U^Q component predicts the original F in the decoding stage particularly in the case of PPHLCT. (Note that in PHLCT we should be satisfied if the U^Q predicts F as well as U does.) From the viewpoint of prediction, there are several algorithms, [1, Sec. 16.1], [13], [14], [18], [15], which predict three to nine low frequency AC components of F. Therefore, in this section, we shall measure the performance of U and U^Q as a predictor of F and compare the results with those obtained by [1, Sec. 16.1] and [13]. We shall not conduct the comparison with the methods proposed by [14], [18], [15] because: 1) the predictors $\widehat{F}_{\boldsymbol{k}}$ for some range of $\boldsymbol{k}$'s proposed by [14] and [18] in the encoding stage require the values of $F_{\boldsymbol{k}}$ themselves and hence the comparison in the encoding stage with our methods does not make sense; 2) the method in [15] needs an optimization by linear programming in order to predict the low frequency AC components of F, in addition to the same problem described in 1).

The goodness of a given predictor $\widehat{F}_{\boldsymbol{k}}$ of $F_{\boldsymbol{k}}$ for a given input image can be measured by the following

quantity:

$$R[\widehat{F}_{\boldsymbol{k}}] := 100 \cdot \left(1 - \frac{\langle |F_{\boldsymbol{k}} - \widehat{F}_{\boldsymbol{k}}| \rangle}{\langle |F_{\boldsymbol{k}}| \rangle}\right), \quad k_1, k_2 = 0, 1, \ldots, 7,$$

where $\langle \cdot \rangle$ denotes the mean value over all blocks in the input image (the outmost blocks are not included). If $\widehat{F}_{\boldsymbol{k}}$ perfectly predicts $F_{\boldsymbol{k}}$, then $R[\widehat{F}_{\boldsymbol{k}}] = 100$.

Table IV shows the prediction performance of various methods for the first 14 DCT coefficients in the zig-zag order after the DC component. Note that this table lists those coefficients not in the usual zig-zag order: it lists the results at the first column and row followed by those at the other indices. In the table, we denote the prediction performance at the encoding stage and the decoding stage by $R_{\boldsymbol{k}}$ and $R_{\boldsymbol{k}}^Q$, respectively. In PHLCT, we set $\widehat{F}_{\boldsymbol{k}} = U_{\boldsymbol{k}}$ in the encoding stage while in the decoding stage $\widehat{F}_{\boldsymbol{k}} = U_{\boldsymbol{k}}^Q$. The performance values in Table IV were obtained from V^Q's compressed at 0.3 bits/pixel (bpp) for both images.

As for PPHLCT, it is only applicable for the decoding stage. Thus, we put "NA" (Not Applicable) in the encoder performance column in the table. Recall that the predictor $\widehat{F}_{\boldsymbol{k}} = U^Q$ in the decoder stage is computed from the quantized original F^Q instead of V^Q in PPHLCT. Hence, the performance should be different from that of the predictor in the decoding stage of PHLCT. In this case, F^Q's were obtained by compressing the original F's at 0.3 bpp, which were also used in the other methods described below.

QSFIT in the table denotes an AC prediction method described in [1, Sec. 16.1]. This method is based on the quadratic surface fitting using the DC components of the current and its eight adjacent blocks. Note that the JPEG standard presents the prediction formulas for the lowest five AC components corresponding to $\boldsymbol{k} \in \{(0,1), (1,0), (2,0), (1,1), (0,2)\}$ in Eq.(16-2a)–Eq.(16-2e) in the encoding stage and Eq.(16-3a)–Eq.(16-3e) in the decoding stage in [1], and hence recommends (as an option) the prediction of these five AC components only. For comparison with our PHLCT and PPHLCT, however, we computed the prediction performance of the first 14 AC components in the zig-zag order by actually fitting the quadratic surface at each block using true DC components in the encoding stage and using the quantized DC components in the decoding stage instead of using the formulas Eq.(16-2a)–Eq.(16-3e).

LAKHANI in the table denotes another AC prediction method proposed in [13] that is also based on the quadratic surface fitting. We computed the prediction performance of the first nine AC components in the zig-zag order using the prediction formulas of Eq.(7)–Eq.(15) in [13]. We note that we could not compute the performance of all the 14 AC components because we could not derive the equation of the quadratic surface from the constraints used in Fig. 3 of [13], and the detailed derivation was not given in that paper.

Table IV shows that the results of QSFIT in the encoding stage for predicting the first column and row of F are exactly the same as those of PHLCT. On the other hand, the QSFIT performance in the decoding stage for those k's is exactly the same as that of PPHLCT. This is because the first column and row of U except the DC component are exactly the same as those of the quadratic surface used in QSFIT. Appendix III describes the detailed reason why these DCT coefficients agree. However, our methods outperform QSFIT for the other AC components. Note that the performance of PHLCT in the decoding stage for some k's is slightly different from that of QSFIT and PPHLCT due to the difference in computing the predictor from V^Q and from F^Q. For the first column and row of F, LAKHANI gives the best results. However, the difference between the performance values before and after quantization is larger than the other methods. For examples, even at $k = (1, 0)$, the performance drop after quantization is more than 10% relative to the true mean value $\langle F_{1,0} \rangle$. Concerning the other range of k's, in particular, at $k = (1, 1), (1, 2), (2, 1)$, our methods and QSFIT perform better than LAKHANI.

B. Image Compression

We now describe our image compression experiments. In addition to the standard LQT with the rule (16), we also introduce a modified version of the rule here. This is based on the following observation from our experiments and experience: the smaller the quantization error of the DC components (regardless of JPEG-DCT, PHLCT, or PPHLCT), the more numerically accurate and the perceptually better the reconstructed images are. In other words, controlling the quantization errors of the DC components is much more important than those of the other higher frequency AC components. Thus, we impose an upper bound on the quantization step size of the DC components in order to keep the accuracy even in the very low bit-rate case. More precisely, the element $Q_{\mathbf{0}}$ in (16) is *modified* as the following rule: *if $Q_{\mathbf{0}} > M$, then replace $Q_{\mathbf{0}}$ by M*. In other words, the quantization step size for the DC components is always bounded by a constant M, whatever the quality factor q is used. In the experiments below, we set the constant M by an ad hoc procedure, e.g., $M = 0.05 \langle F_{\mathbf{0}} \rangle$, where $\langle F_{\mathbf{0}} \rangle$ denotes the mean value of the DC components of all the blocks of size 8×8 pixels in an input image. We shall denote this modified quantization rule by QM whereas the standard rule (16) by QS.

To measure the performance of compression and decompression objectively, we use the Peak Signal-to-Noise Ratio (PSNR) and the Mean Structural SIMilarity index (MSSIM). PSNR is normally considered a better metric for image quality assessment than SNR and defined as

$$20 \log_{10} \left(\max_{(x,y) \in \Omega} |f^Q(x, y)| / RMSE \right),$$

TABLE IV

COMPARISON WITH OTHER AC PREDICTION METHODS

k	R_k for Gabor				R_k for Lenna				R_k for Barbara			
	PHLCT	PPHLCT	QSFIT	LAKHANI	PHLCT	PPHLCT	QSFIT	LAKHANI	PHLCT	PPHLCT	QSFIT	LAKHANI
(0, 1)	61	NA	61	81	28	NA	28	47	31	NA	31	41
(1, 0)	61	NA	61	81	33	NA	33	50	34	NA	34	47
(2, 0)	76	NA	76	58	13	NA	13	21	12	NA	12	19
(0, 2)	76	NA	76	58	13	NA	13	20	12	NA	12	18
(0, 3)	78	NA	78	87	8.6	NA	8.6	9.3	6.2	NA	6.2	5.3
(3, 0)	78	NA	78	87	8.4	NA	8.4	8.8	6.5	NA	6.5	7.2
(4, 0)	78	NA	78	NA	5.7	NA	5.7	NA	5.9	NA	5.9	NA
(0, 4)	78	NA	78	NA	5.7	NA	5.7	NA	2.5	NA	2.5	NA
(1, 1)	62	NA	39	16	11	NA	2.2	12	12	NA	3.4	−23
(1, 2)	72	NA	47	−67	6.2	NA	0.6	−6.6	3.8	NA	0.9	−25
(2, 1)	72	NA	47	−67	6.9	NA	−0.1	−24	4.8	NA	0.7	−36
(3, 1)	73	NA	47	NA	2.8	NA	0.8	NA	1.3	NA	0.4	NA
(2, 2)	64	NA	50	NA	4.1	NA	0.1	NA	1.2	NA	0.2	NA
(1, 3)	73	NA	47	NA	1.9	NA	0.5	NA	0.3	NA	0.1	NA
k	R_k^Q for Gabor				R_k^Q for Lenna				R_k^Q for Barbara			
	PHLCT	PPHLCT	QSFIT	LAKHANI	PHLCT	PPHLCT	QSFIT	LAKHANI	PHLCT	PPHLCT	QSFIT	LAKHANI
(0, 1)	61	61	61	79	28	28	28	43	30	30	30	34
(1, 0)	61	61	61	79	32	32	32	42	33	33	33	36
(2, 0)	75	74	74	53	12	12	12	17	12	11	11	14
(0, 2)	75	74	74	54	12	12	12	19	12	12	12	15
(0, 3)	78	78	78	80	8.5	8.5	8.5	8.8	6.2	6.1	6.1	4.6
(3, 0)	78	78	78	80	8.4	8.2	8.2	7.4	6.4	6.3	6.3	5.7
(4, 0)	77	76	76	NA	5.5	5.4	5.4	NA	5.6	5.7	5.7	NA
(0, 4)	77	76	76	NA	5.7	5.6	5.6	NA	2.4	2.5	2.5	NA
(1, 1)	63	61	39	22	11	11	2.1	4.0	12	12	3.3	−37
(1, 2)	72	69	47	−79	6.0	5.9	0.6	−14	4.2	3.5	0.9	−40
(2, 1)	72	68	47	−78	6.4	6.6	−0.1	−32	4.6	4.5	0.6	−51
(3, 1)	73	71	47	NA	2.7	2.7	0.8	NA	1.4	1.4	0.4	NA
(2, 2)	65	56	49	NA	3.8	3.9	0.1	NA	1.4	1.3	0.1	NA
(1, 3)	73	70	47	NA	1.9	1.9	0.5	NA	0.4	0.4	0.1	NA

where Ω is the entire image domain (not a single block of 8×8 pixels) and $RMSE$ (the root mean square error) is the absolute ℓ^2 error between the original image and the reconstruction divided by the square root of the total number of pixels. The unit of PSNR is decibel (dB). MSSIM is a more perceptually-correct measure of similarity between two images that was recently proposed by Wang et al. [26]. This is based on the comparison of the local patterns of pixel intensities normalized for luminance and contrast. If the original image is regarded as the one with the perfect quality, then the MSSIM between the original and the reconstruction can be considered as a quality measure of the reconstructed image. Note that the value of MSSIM becomes 1 if the reconstruction perfectly recovers the original.

Figures 3 and 4 show the comparison between PHLCT, PPHLCT, and the JPEG Baseline method for 512×512 grayscale images of Gabor, Lenna, and Barbara, respectively. In these figures, the quality factor q in (16) was sampled uniformly with the sampling rate $\Delta q = 3$ so that the bit rates take values approximately from 0.15 bpp up to 1.0 bpp. Then, the images were compressed accordingly using the rules QM and QS in all cases. The bit rates were computed from the Huffman codes of the quantized data in the zig-zag order. Finally, PSNR and MSSIM were computed after decompressing them. We note that in order to obtain the results indicated by QM, not only the decoder but also the encoder must use the QM rule. Therefore, PPHLCT with QM is effective only when the QM rule has been used in the encoder of the JPEG Baseline method.

In Figures 3 and 4, PHLCT(QM) is the overall winner, and the improvement over other methods by PHLCT and PPHLCT is more clearly shown in MSSIM than in PSNR. In all cases, the performance difference between QS and QM is particularly noticeable for the low bit-rate range smaller than 0.3 bpp. On the other hand, if one can afford to use more than 0.3 bpp, there seems no need to use QM. The improvement by PPHLCT(QS) is the smallest among our proposed methods, which was also expected since this is the only case that does not modify the encoder of the JPEG Baseline method at all. The improvement by PPHLCT(QM) is slightly larger than that by PPHLCT(QS) while it is still smaller than that by PHLCT(QS). In addition, these figures also indicate that it is tougher to compress the Barbara image than the other images.

Next, we show the performance difference between PHLCT, QSFIT, and LAKHANI using PSNR. The AC predictors $\widehat{F}$ of QSFIT and LAKHANI were applied in the same manner as PHLCT. Only the residuals $V = F - \widehat{F}$ were coded at the encoding stage. We computed $\widehat{F}$ from V^Q and reconstructed $F^Q = \widehat{F} + V^Q$ at the decoding stage. As for the "partial mode" versions of QSFIT and LAKHANI, which are denoted by PQSFIT and PLAKHANI, $\widehat{F}$ were used in the same manner as PPHLCT only for the decoding stage, such that

$$\text{Replace } F_{\boldsymbol{k}}^Q \text{ by } \widehat{F}_{\boldsymbol{k}} \quad \text{if } \boldsymbol{k} \in \mathcal{K}_t,$$

$$\mathcal{K}_t := \left\{ \boldsymbol{k} \in \mathcal{K} \ \middle| \ F_{\boldsymbol{k}}^Q = 0 \text{ and } |\widehat{F}_{\boldsymbol{k}}| < Q_{\boldsymbol{k}}/2 \right\}.$$

Figure 5 shows the PSNR gain by PHLCT, PPHLCT, QSFIT, PQSFIT, LAKHANI, and PLAKHANI over the JPEG Baseline method for the bit rates from 0.15 bpp up to 1.0 bpp. In Figure 5, it is clear that the gain by PHLCT is the largest in all cases and the gain by PPHLCT is larger than PQSFIT and PLAKHANI. The gain by QSFIT (full mode) is larger than PPHLCT (partial mode) for the Lenna and Barbara images, which is understandable. However, for the Gabor image, the gain by PPHLCT is larger than the full
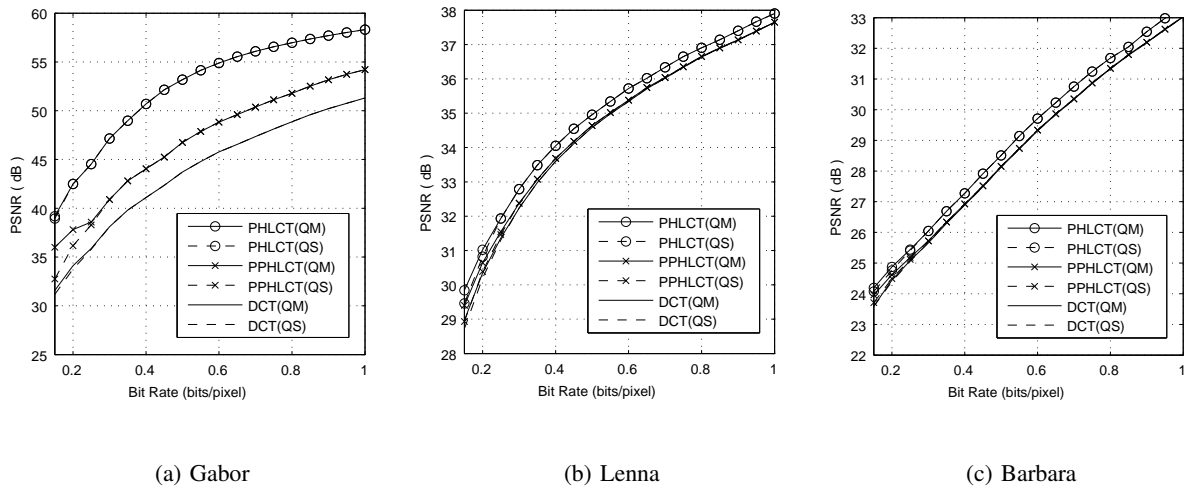
(a) Gabor (b) Lenna (c) Barbara

Fig. 3. Comparison of the compression performance with PSNR between PHLCT, PPHLCT, and DCT(JPEG Baseline method).

mode versions of QSFIT and LAKHANI in the range of 0.4–1.0 bpp. In each method, the results of full mode version are better than those of partial mode version except LAKHANI/PLAKHANI. The reason why the gain by LAKHANI is smaller than PLAKHANI seems to be due to the large difference between the AC prediction performance before and after quantization compared to the other methods.

We also conducted our experiments on six more images shown in Figure 6 in order to test our algorithms over a variety of images with different characteristics. Figure 7 shows the PSNR gain by PHLCT(QM), PPHLCT(QM), QSFIT(QM), PQSFIT(QM), LAKHANI(QM), and PLAKHANI(QM) for those six images. Note that we only show the results with QM in order to save the space; the performance with QS is inferior to that with QM similarly to our results shown in Figure 5. At least for these test images, PHLCT(QM) was over all the winner among the full versions of the methods we tried, so was PPHLCT(QM) among the partial versions.

Finally, to show the effect of the compression-decompression quality of our methods more clearly, we display the center part of the reconstructed Gabor images in Figures 8−9 and the face region of the reconstructed Lenna and Barbara images in Figures 10−13. Note that the bit rates and the PSNR values listed with the images are computed from the entire images. The quality factors that achieved those bit rates and PSNR values are also listed with the images. It is clear that both PHLCT and PPHLCT improved the quality of the JPEG-compressed images under the constraint of keeping the same bit rates. The difference is particularly noticeable for the low bit-rate case (0.15 bpp).
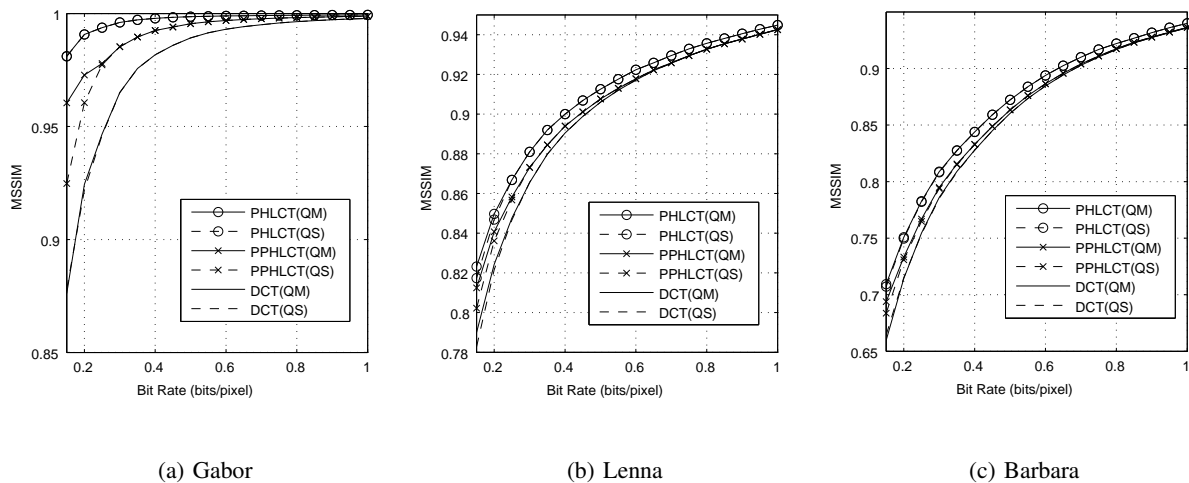
(a) Gabor   (b) Lenna   (c) Barbara

Fig. 4. Comparison of the compression performance with MSSIM between PHLCT, PPHLCT, and DCT(JPEG Baseline method).

## C. Quantification of Blocking Artifacts Reduction

We also evaluated the performance of the reduction of blocking artifacts in the decompressed images by using a measure called the *mean squared difference of slope* (MSDS) proposed in [27] and used (with modification) in [28] and [18]. The MSDS values were computed from the sequences of four consecutive pixel values in horizontal/vertical directions across a block boundary or of four pixel values in diagonal directions at an intersection of diagonally adjacent blocks as in [18]. We computed MSDS for all block boundaries and all intersections in each image.

Table V shows our computational results of MSDS for the Gabor, Lenna, and Barbara images compressed by PHLCT(QM), PPHLCT(QM), QSFIT(QM), PQSFIT(QM), LAKHANI(QM), PLAKHANI(QM), and DCT(QM). In this table, MSDSb and MSDSi denote the mean value of MSDS computed at all the block boundaries and that computed at all the intersections, respectively. We can observe several things from this table. First of all, this table demonstrates that PHLCT is the best (i.e., smallest in terms of MSDS values) among all the full version methods we examined, so is PPHLCT among all the partial versions.

Second, the performance difference between PHLCT and the other methods for the Gabor image is particularly noticeable. This is understandable because the original Gabor image do not have any intrinsic discontinuity and PHLCT efficiently reduces the code size of such an image with less artifacts thanks to its fast decaying DCT coefficients of the residual as Corollary 4 suggests.
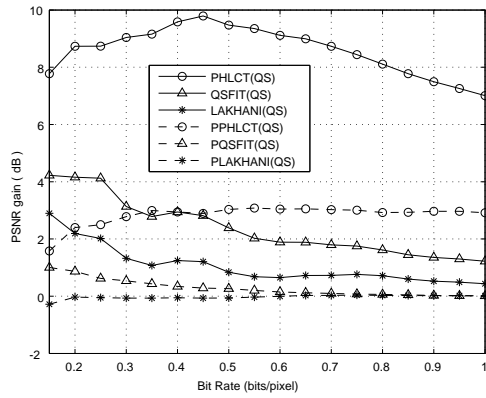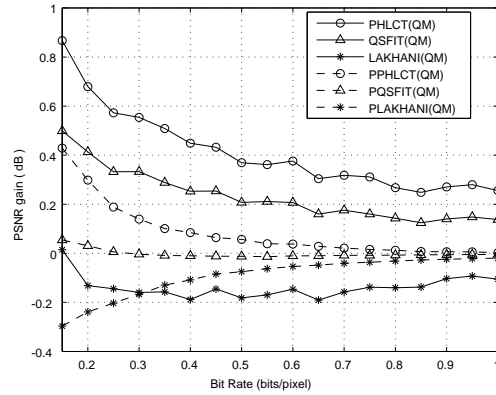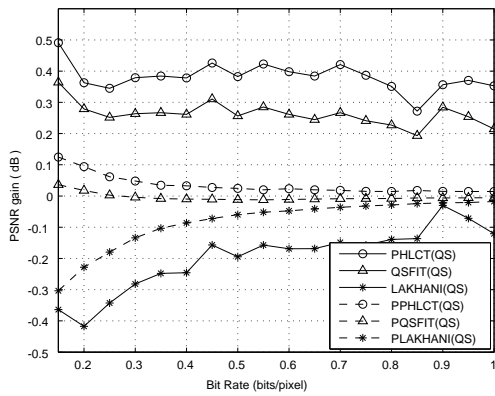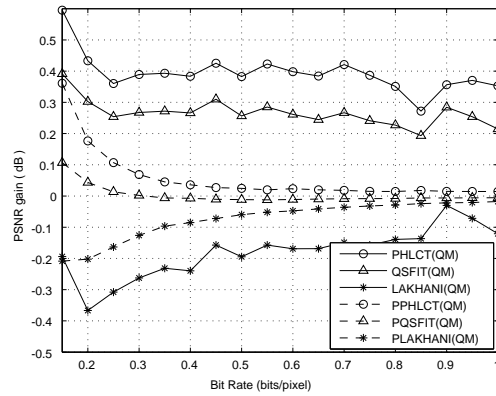
(a) Gabor, QS
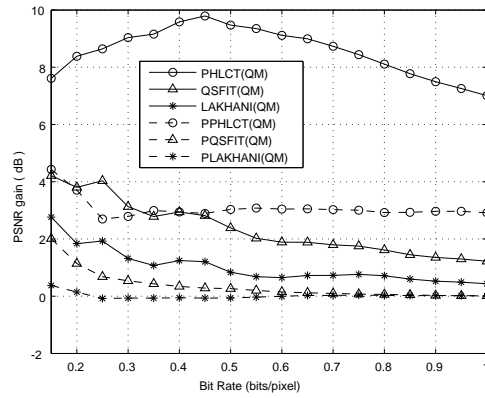
(b) Gabor, QM
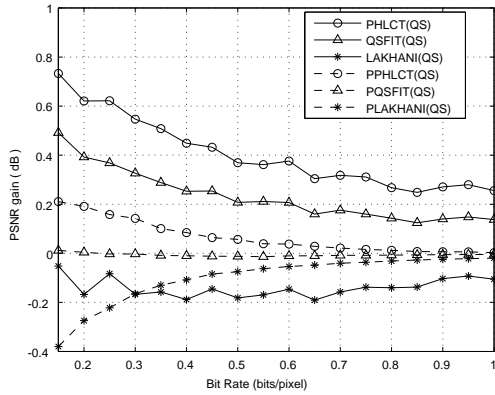
(c) Lenna, QS

(d) Lenna, QM

(e) Barbara, QS

(f) Barbara, QM

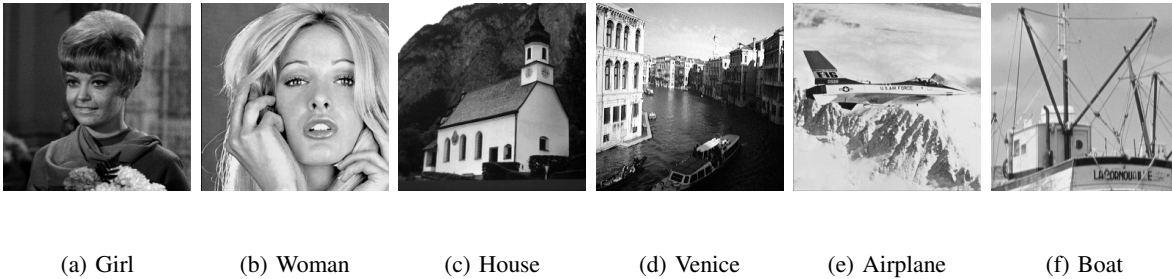Fig. 5. PSNR gain by PHLCT, PPHLCT, QSFIT, PQSFIT, LAKHANI, and PLAKHANI over the JPEG Baseline method.

| (a) Girl | (b) Woman | (c) House | (d) Venice | (e) Airplane | (f) Boat |

Fig. 6.   Test images (256×256 pixels, 8 bits/pixel grayscale).

TABLE V

MSDS OF RECONSTRUCTED IMAGES

|		Gabor			Lenna			Barbara		
		MSDSb	MSDSi	PSNR	MSDSb	MSDSi	PSNR	MSDSb	MSDSi	PSNR
	Original	3	2		645	379		5190	3082	
0.15 bpp	PHLCT(QM)	105	40	39.21	1372	889	29.84	6120	2893	24.19
	QSFIT(QM)	863	352	35.79	2311	1250	29.46	7831	3371	23.98
	LAKHANI(QM)	311	383	34.34	1742	1163	28.97	7547	3850	23.37
	PPHLCT(QM)	282	125	35.69	1452	932	29.38	5743	2657	23.97
	PQSFIT(QM)	1245	290	33.37	2386	1275	29.00	7589	3233	23.73
	PLAKHANI(QM)	1107	458	31.96	2257	1374	28.67	7232	3564	23.41
	DCT(QM)	2004	710	31.41	2658	1455	28.93	8360	3593	23.61
0.3 bpp	PHLCT(QM)	30	11	47.02	1062	670	32.80	9113	4121	26.05
	QSFIT(QM)	253	135	41.26	1376	817	32.57	10119	4426	25.93
	LAKHANI(QM)	193	156	39.41	1230	797	32.08	9908	4663	25.38
	PPHLCT(QM)	62	28	40.89	1079	689	32.36	9052	4154	25.73
	PQSFIT(QM)	351	181	38.68	1426	831	32.22	10047	4454	25.67
	PLAKHANI(QM)	348	228	38.05	1393	851	32.06	9996	4546	25.54
	DCT(QM)	417	223	38.12	1470	862	32.22	10203	4522	25.67

Finally, the MSDS values for the Barbara images reconstructed from the compressed representations of 0.3 bpp are larger than those of 0.15 bpp. This is because the intrinsic discontinuities and textures in the original Barbara image had been too much smoothed in the case of 0.15 bpp. However, for each bit rate, the MSDS values of PHLCT are still smaller than the other full version methods, so are those of PPHLCT than the other partial versions. Therefore, we can conclude that both PHLCT and PPHLCT are effective for the reduction of blocking artifacts in terms of MSDS.
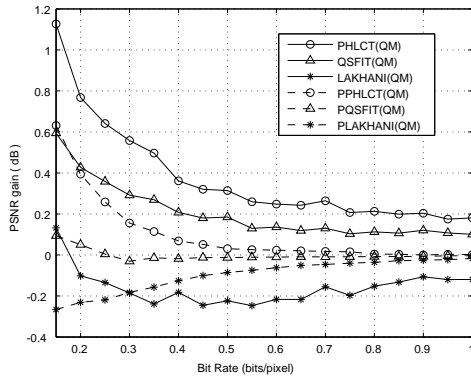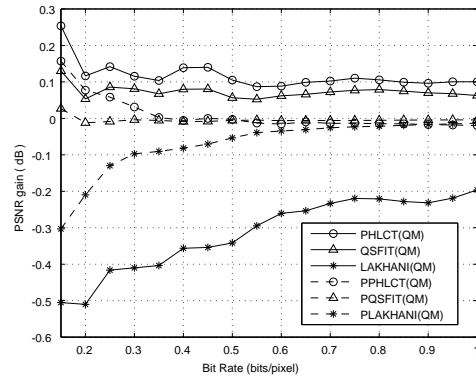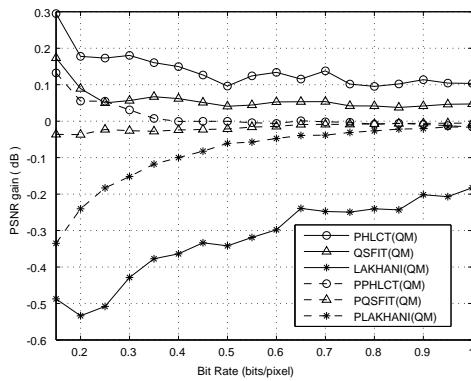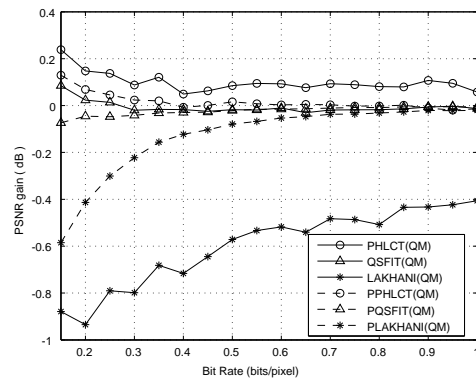
(a) Girl, QM
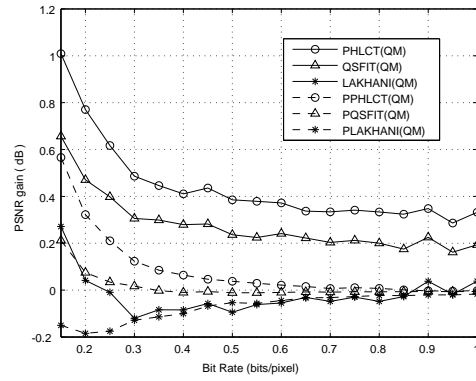
(b) Woman, QM
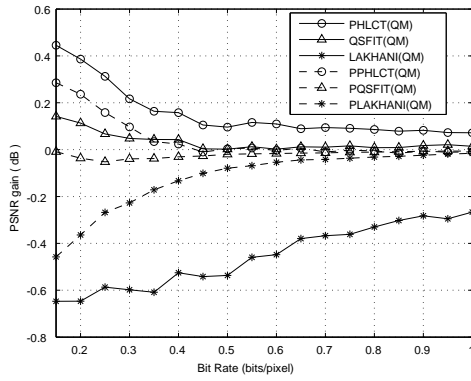
(c) House, QM

(d) Venice, QM

(e) Airplane, QM

(f) Boat, QM

Fig. 7. PSNR gain by PHLCT(QM), PPHLCT(QM), QSFIT(QM), PQSFIT(QM), LAKHANI(QM), and PLAKHANI(QM) over the JPEG Baseline method for various test images.

(a) PHLCT(QS), $q =$ 15.07, 39.12 dB

(b) QSFIT(QS), $q =$ 12.16, 35.30 dB

(c) PPHLCT(QS), $q = 6.93$, 32.89 dB

(d) PQSFIT(QS), $q = 6.93$, 32.22 dB

(e) DCT(QS), $q =$ 6.93, 31.17 dB

(f) PHLCT(QM), $q = 14.90$, 39.21 dB

(g) QSFIT(QM), $q =$ 11.57, 35.79 dB

(h) PPHLCT(QM), $q = 5.50$, 35.69 dB

(i) PQSFIT(QM), $q = 5.50$, 33.37 dB
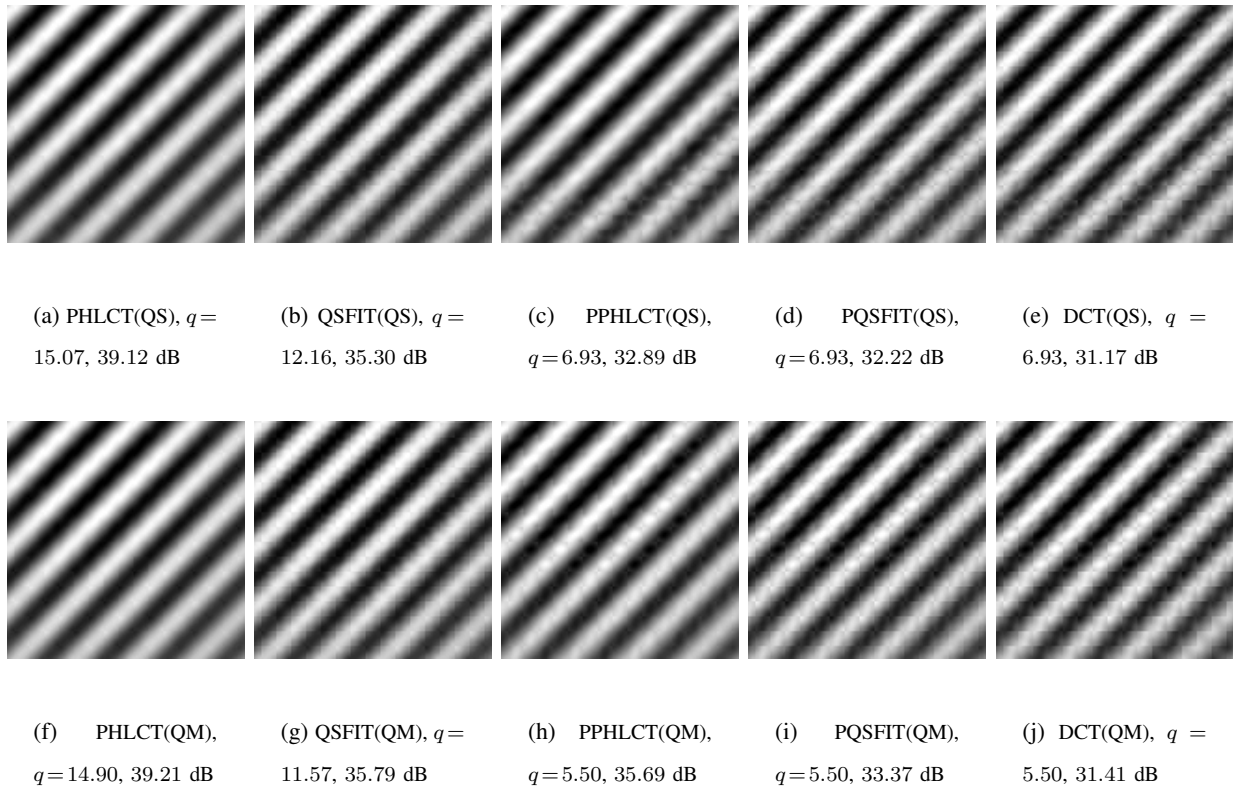
(j) DCT(QM), $q =$ 5.50, 31.41 dB

Fig. 8. The center part of the reconstructed Gabor images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.15 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the center part.

VIII. CONCLUSION

In this paper, we have described two new image compression-decompression schemes that reproduce images with better visual fidelity, less blocking artifacts, and better PSNR, particularly in low bit rates, than those processed by the JPEG Baseline method at the same bit rates. The first one, the "full mode" polyharmonic local cosine transform (or PHLCT for short), modifies both the encoder and decoder parts of the JPEG Baseline method. The second one, the "partial mode" PHLCT (or PPHLCT for short), only modifies the decoder part: it accepts the JPEG files, yet decompresses them with higher quality than the JPEG standard. The key idea behind these algorithms is a decomposition of each image block into a *polyharmonic* component and a *residual*. The polyharmonic component in this paper is an approximate solution to Poisson's equation with the Neumann boundary condition, which can be considered as a smooth predictor of the original image block using the image gradient information across the block boundary. The residual—obtained by removing the polyharmonic component from the original image

(a) PHLCT(QS), $q=$ 52.0, 47.02 dB

(b) QSFIT(QS), $q=$ 34.1, 41.26 dB

(c) PPHLCT(QS), $q=20.5$, 40.89 dB

(d) PQSFIT(QS), $q=20.5$, 38.68 dB

(e) DCT(QS), $q=$ 20.5, 38.12 dB

(f) PHLCT(QM), $q=52.0$, 47.02 dB

(g) QSFIT(QM), $q=$ 34.1, 41.26 dB

(h) PPHLCT(QM), $q=20.5$, 40.89 dB

(i) PQSFIT(QM), $q=20.5$, 38.68 dB

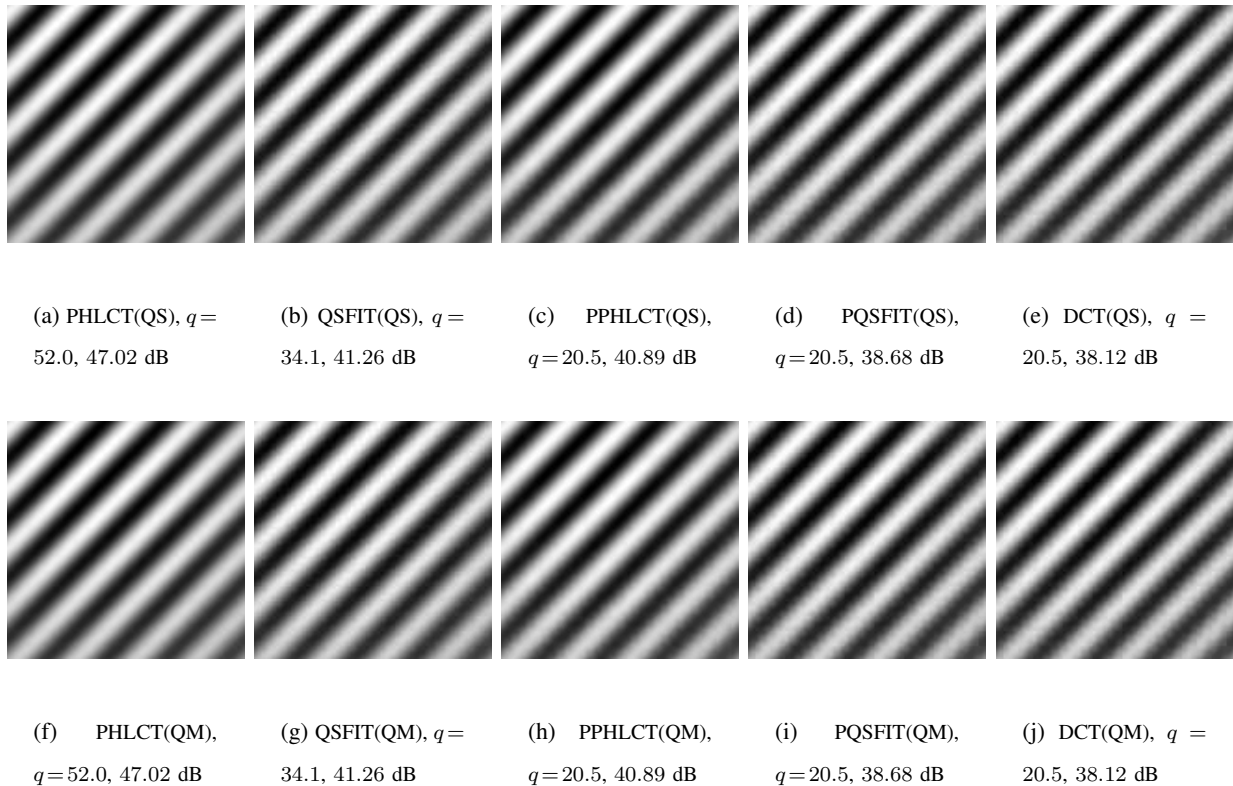(j) DCT(QM), $q=$ 20.5, 38.12 dB

Fig. 9. The center part of the reconstructed Gabor images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.3 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the center part.

block—has thus approximately zero gradient across the block boundary, which gives rise to the DCT coefficients decaying as $O(\|\boldsymbol{k}\|^{-4})$ whereas those of the original image block decay only as $O(\|\boldsymbol{k}\|^{-2})$. This fast decay rate in turn has led us to more efficient compression-decompression algorithms for the same bit rates. Moreover, we have shown that the polyharmonic component of each block can be estimated solely by the first column and row of the DCT coefficient matrix of that block and those of its adjacent blocks. This is important in two aspects: 1) it provides a simpler decompression algorithm; and 2) it allows us to develop PPHLCT that fills in the truncated small DCT coefficients of the original image due to quantization with the corresponding polyharmonic components.

The additional computational cost incurred by our new methods is small. In addition to the cost of the JPEG Baseline method, both the encoder and decoder parts of PHLCT requires $O(N^2)$ operations per block of size $N \times N$ pixels and $O(N^2 \log_2 N)$ operations to compute Table I or Eq. (10) per image, where typically $N = 8$. The cost of computing Table I is essentially negligible since this is done only

(a) PHLCT(QS), $q=$ 8.39, 29.50 dB

(b) QSFIT(QS), $q=$ 8.13, 29.22 dB

(c) PPHLCT(QS), $q=7.35$, 28.89 dB

(d) PQSFIT(QS), $q=7.35$, 28.70 dB

(e) DCT(QS), $q=$ 7.35, 28.70 dB

(f) PHLCT(QM), $q=7.34$, 29.84 dB

(g) QSFIT(QM), $q=$ 7.10, 29.46 dB

(h) PPHLCT(QM), $q=6.25$, 29.38 dB

(i) PQSFIT(QM), $q=6.25$, 29.00 dB

(j) DCT(QM), $q=$ 6.25, 28.93 dB

Fig. 10. The face part of the reconstructed Lenna images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.15 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the face part.

once in the encoder as well as in the decoder. As for PPHLCT, it has only a decoder part and requires $O(3N^2)$ comparison operations per block in addition to the cost of the decoder part of PHLCT. Hence we can claim that the additional cost incurred by our methods beyond the JPEG Baseline method is just linearly proportional to the number of pixels in the image.

Our numerical experiments in Section VII have demonstrated that PHLCT and PPHLCT improve the decompressed image quality over the JPEG Baseline method under the constraint of keeping the same bit rates. Here we have used PSNR and MSSIM as a measure of image quality and MSDS to specifically quantify the reduction of blocking artifacts. The subjective image quality is also improved by our methods as shown in Figures 10−13. We have also numerically demonstrated that the polyharmonic component of our methods can predict an original image data better and more stably than the other AC prediction methods such as [1, Sec. 16.1] and [13].

Our future work will include a more thorough investigation of polyharmonic components that may be

(a) PHLCT(QS), $q =$ 20.30, 32.80 dB

(b) QSFIT(QS), $q =$ 19.60, 32.57 dB

(c) PPHLCT(QS), $q = 17.95$, 32.36 dB

(d) PQSFIT(QS), $q = 17.95$, 32.22 dB

(e) DCT(QS), $q =$ 17.95, 32.22 dB
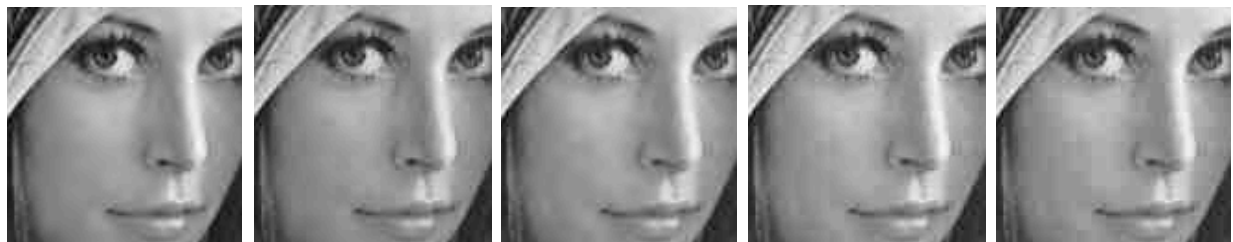
(f) PHLCT(QM), $q = 20.30$, 32.80 dB

(g) QSFIT(QM), $q =$ 19.60, 32.57 dB

(h) PPHLCT(QM), $q = 17.95$, 32.36 dB

(i) PQSFIT(QM), $q = 17.95$, 32.22 dB

(j) DCT(QM), $q =$ 17.95, 32.22 dB

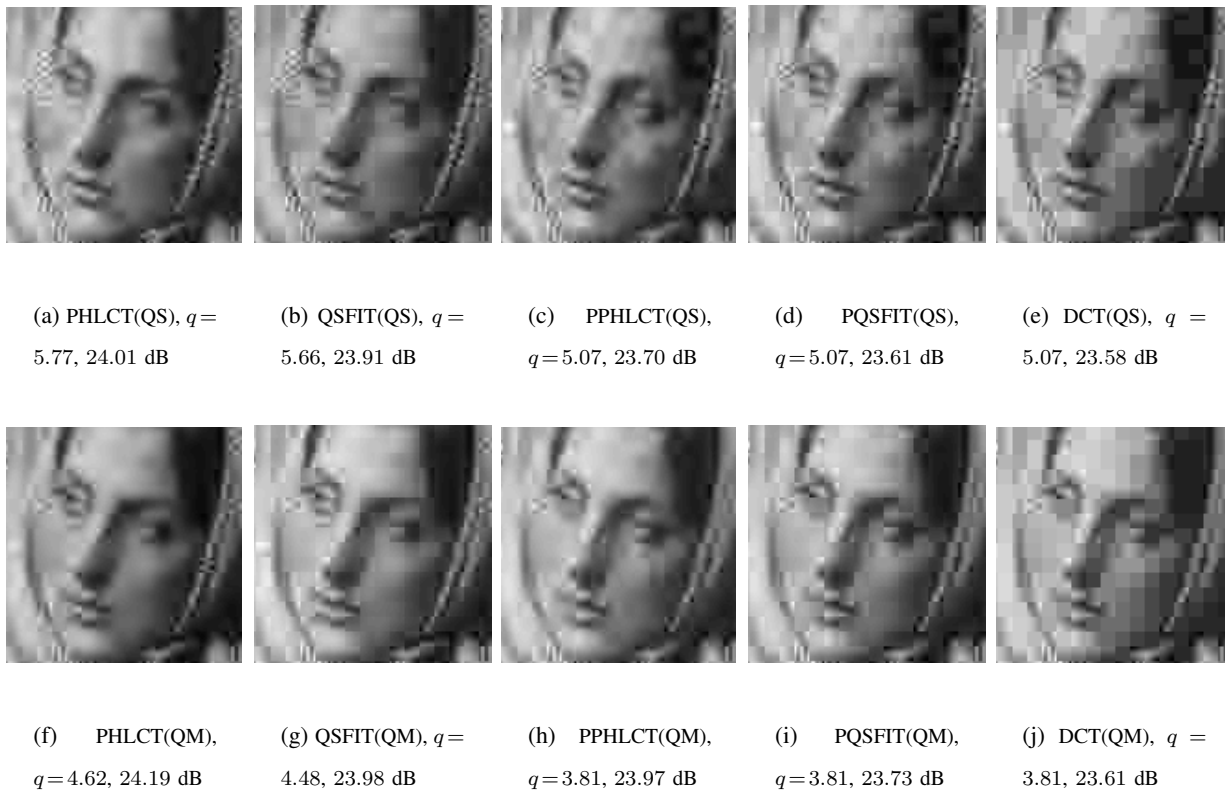Fig. 11. The face part of the reconstructed Lenna images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.3 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the face part.

obtained by a polyharmonic equation of higher degree of polyharmonicity or by a different type of elliptic equation. Furthermore, we are currently generalizing the polyharmonic local trigonometric transform for analyzing objects of various shapes in an image, which we hope to report at a later date.

APPENDIX I

ON THE NEUMANN BOUNDARY CONDITION

The reason why we cannot use Laplace's equation with the Neumann boundary condition in general is due to Green's second identity, which claims that for any $u, v \in C^1(\overline{\Omega})$,

$$\int_\Omega (u\Delta v - v\Delta u) \, d\boldsymbol{x} = \int_{\partial\Omega} \left(u\frac{\partial v}{\partial \nu} - v\frac{\partial u}{\partial \nu} \right) \, d\sigma(\boldsymbol{x}), \qquad (20)$$

(a) PHLCT(QS), $q =$ 5.77, 24.01 dB

(b) QSFIT(QS), $q =$ 5.66, 23.91 dB

(c) PPHLCT(QS), $q = 5.07$, 23.70 dB

(d) PQSFIT(QS), $q = 5.07$, 23.61 dB

(e) DCT(QS), $q =$ 5.07, 23.58 dB

(f) PHLCT(QM), $q = 4.62$, 24.19 dB

(g) QSFIT(QM), $q =$ 4.48, 23.98 dB

(h) PPHLCT(QM), $q = 3.81$, 23.97 dB

(i) PQSFIT(QM), $q = 3.81$, 23.73 dB

(j) DCT(QM), $q =$ 3.81, 23.61 dB

Fig. 12. The face part of the reconstructed Barbara images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.15 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the face part.

where $d\sigma(\boldsymbol{x})$ is a surface (or boundary) measure. Setting $v = 1$ in (20) together with the Neumann boundary condition in (3), we have

$$\int_\Omega \Delta u \, d\boldsymbol{x} = \int_{\partial\Omega} \frac{\partial u}{\partial \nu} \, d\sigma(\boldsymbol{x}) = \int_{\partial\Omega} \frac{\partial f}{\partial \nu} \, d\sigma(\boldsymbol{x}). \tag{21}$$

This is a necessary condition that u must satisfy. Therefore, we need to introduce the constant source term in Laplace's equation (3) as

$$\begin{cases} \Delta u = K & \text{in } \Omega, \\ \dfrac{\partial u}{\partial \nu} = \dfrac{\partial f}{\partial \nu} & \text{on } \partial\Omega, \end{cases} \tag{4}$$

where $K := \frac{1}{|\Omega|} \int_{\partial\Omega} \partial f / \partial \nu \, d\sigma(\boldsymbol{x})$, $|\Omega|$ is the volume of the block Ω. It is easy to see that with this constant source term, the solution of (4) satisfies (21).
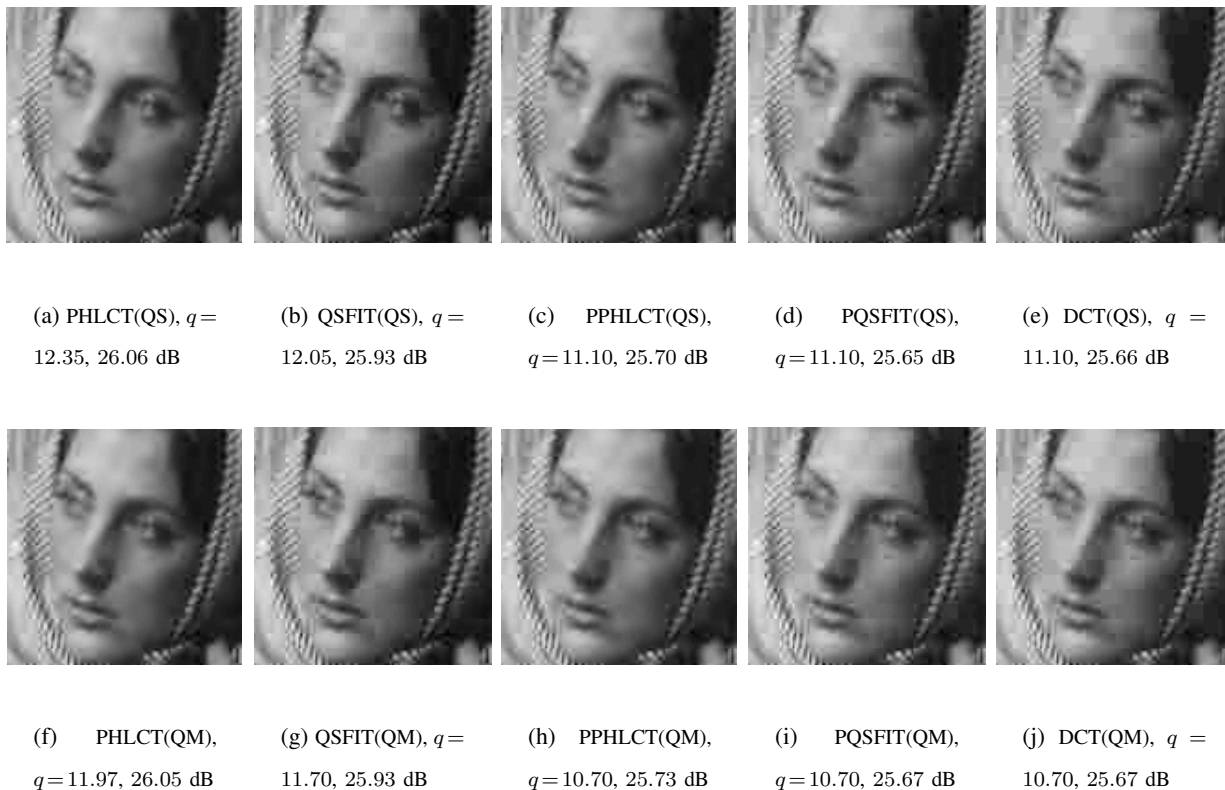
(a) PHLCT(QS), $q=$ 12.35, 26.06 dB

(b) QSFIT(QS), $q=$ 12.05, 25.93 dB

(c) PPHLCT(QS), $q=11.10$, 25.70 dB

(d) PQSFIT(QS), $q=11.10$, 25.65 dB

(e) DCT(QS), $q=$ 11.10, 25.66 dB

(f) PHLCT(QM), $q=11.97$, 26.05 dB

(g) QSFIT(QM), $q=$ 11.70, 25.93 dB

(h) PPHLCT(QM), $q=10.70$, 25.73 dB

(i) PQSFIT(QM), $q=10.70$, 25.67 dB

(j) DCT(QM), $q=$ 10.70, 25.67 dB

Fig. 13. The face part of the reconstructed Barbara images by PHLCT, QSFIT, PPHLCT, PQSFIT, and DCT(JPEG Baseline method) with 0.3 bpp. The bit rates and the PSNR values listed with the images are computed from the entire image, not from the face part.

APPENDIX II

PROOF OF THEOREM 3

Proof: Recall that u is the solution to the Neumann boundary problem of Poisson's equation (4) (modulo an additive constant).

$$
\begin{aligned}
M(p) - M(u) &= \int_\Omega \left((\Delta p)^2 - (\Delta u)^2 \right) \, d\boldsymbol{x} \\
&= \int_\Omega (\Delta p - \Delta u)^2 \, d\boldsymbol{x} + 2 \int_\Omega (\Delta p - \Delta u) \Delta u \, d\boldsymbol{x} \\
&= \int_\Omega (\Delta p - \Delta u)^2 \, d\boldsymbol{x} + 2K \int_\Omega (\Delta p - \Delta u) \, d\boldsymbol{x} \, .
\end{aligned}
$$

Using Green's second identity, we have

$$
\int_\Omega (\Delta p - \Delta u) \, d\boldsymbol{x} = \int_{\partial\Omega} \frac{\partial p}{\partial \nu} \, d\sigma(\boldsymbol{x}) - \int_{\partial\Omega} \frac{\partial u}{\partial \nu} \, d\sigma(\boldsymbol{x}) = 0 \, .
$$

Therefore

$$M(p) - M(u) = \int_{\Omega} (\Delta p - \Delta u)^2 \, \mathrm{d}\boldsymbol{x} \geq 0 \, .$$

$\blacksquare$

APPENDIX III

EQUIVALENCE OF THE FIRST COLUMN AND ROW OF U AND THOSE OF THE QUADRATIC SURFACE

USED IN QSFIT

As shown in [1, Sec. 16.1], QSFIT uses a quadratic surface

$$p(x, y) = A_1 x^2 y^2 + A_2 x^2 y + A_3 xy^2 + A_4 x^2$$

$$+ A_5 xy + A_6 y^2 + A_7 x + A_8 y + A_9.$$

The coefficients $A_1, \ldots, A_9$ are determined by requiring that the DC components of the current (or center) block and its surrounding eight blocks of an input image match those of $p(x, y)$ when $p(x, y)$ is fitted to these nine blocks.

First of all, we show that the first column of U in (13) except the DC component are exactly the same as those of the DCT coefficients of $p(x, y)$ at the current block. Let P_{k_1, k_2} be the DCT coefficients of $p(x, y)$ on the current block Ω. The first column of P_{k_1, k_2} can be expressed as

$$P_{k_1, 0} = \frac{\sqrt{2}}{N} \sum_{j=0}^{N-1} \left(\lambda_{k_1} \sum_{i=0}^{N-1} p(x_i, \, y_j) \cos \pi k_1 x_i \right)$$

$$= \frac{\sqrt{2}}{N} \lambda_{k_1} \sum_{i=0}^{N-1} \left(\sum_{j=0}^{N-1} p(x_i, \, y_j) \right) \cos \pi k_1 x_i$$

$$= \frac{\sqrt{2}}{N} \lambda_{k_1} \sum_{i=0}^{N-1} \left(ax_i^2 + bx_i + c \right) \cos \pi k_1 x_i,$$

where

$$a = \sum_{j=0}^{N-1} (A_1 y_j^2 + A_2 y_j + A_4),$$

$$b = \sum_{j=0}^{N-1} (A_3 y_j^2 + A_5 y_j + A_7),$$

$$c = \sum_{j=0}^{N-1} (A_6 y_j^2 + A_8 y_j + A_9).$$

Using a, b, and c, the DC component $P_{0,0}$ is expressed by

$$P_{0,0} = \frac{1}{N} \sum_{i=0}^{N-1} \left(ax_i^2 + bx_i + c \right)$$

$$= \frac{4N^2 - 1}{12N^2} a + \frac{1}{2} b + c.$$

Let $P_{0,0}^{(-1,0)}$ and $P_{0,0}^{(1,0)}$ be the DC components of $p(x,y)$ on the adjacent blocks $\Omega^{(-1,0)}$ and $\Omega^{(1,0)}$, respectively. Similarly, we can derive the following DC components of $p(x,y)$ at the adjacent blocks:

$$P_{0,0}^{(-1,0)} = \frac{4N^2 - 1}{12N^2} a - \frac{1}{2} b + c,$$

$$P_{0,0}^{(1,0)} = \frac{28N^2 - 1}{12N^2} a + \frac{3}{2} b + c.$$

Solving these equations for a and b, the first column of P_{k_1,k_2} (except the DC component) can be written by

$$\begin{aligned}
P_{k_1,0} &= \frac{\sqrt{2}}{N} \lambda_{k_1} \sum_{i=0}^{N-1} \left\{ \left(P_{0,0}^{(1,0)} + P_{0,0}^{(-1,0)} - 2P_{0,0} \right) x_i^2/2 \right. \\
&\quad \left. + \left(P_{0,0} - P_{0,0}^{(-1,0)} \right) x_i + c \right\} \cos \pi k_1 x_i \\
&= \left(P_{0,0}^{(-1,0)} - P_{0,0} \right) \frac{\sqrt{2}}{N} \lambda_{k_1} \sum_{i=0}^{N-1} \frac{(x_i - 1)^2}{2} \cos \pi k_1 x_i \\
&\quad + \left(P_{0,0}^{(1,0)} - P_{0,0} \right) \frac{\sqrt{2}}{N} \lambda_{k_1} \sum_{i=0}^{N-1} \frac{x_i^2}{2} \cos \pi k_1 x_i \\
&= \left(P_{0,0}^{(-1,0)} - P_{0,0} \right) \frac{\eta_{0,k_1}}{\sqrt{N}} + \left(P_{0,0}^{(1,0)} - P_{0,0} \right) \frac{\eta_{0,k_1}^*}{\sqrt{N}},
\end{aligned} \tag{22}$$

where we used $\sum_{i=0}^{N-1} \cos \pi k_1 x_i = 0$ to derive the second equality and the definition of η, η^* in (10) to derive the last equality. Now, the DC components of $p(x,y)$ on Ω, $\Omega^{(-1,0)}$, and $\Omega^{(1,0)}$ must match those of an input image in QSFIT. This means that we have $P_{0,0} = F_{0,0}$, $P_{0,0}^{(-1,0)} = F_{0,0}^{(-1,0)}$, and $P_{0,0}^{(1,0)} = F_{0,0}^{(1,0)}$. Putting these into (22) and comparing it with (13), we obtain

$$P_{k_1,0} = U_{k_1,0}, \quad k_1 = 1, 2, \ldots, N-1.$$

Similarly to the case of the first column described above, it is also straightforward to show that the first row of U except the DC component are the same as those of P i.e.,

$$P_{0,k_2} = U_{0,k_2}, \quad k_2 = 1, 2, \ldots, N-1.$$

ACKNOWLEDGMENT

REFERENCES

[1] W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*. New York: Van Nostrand Reinhold, 1993.

[2] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consumer Electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[3] N. Ahmed, T. Natarayan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. COM-23, pp. 90–93, 1974.

[4] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, and Applications*. San Diego, CA: Academic Press, 1990.

[5] G. Strang, "The discrete cosine transform," *SIAM Review*, vol. 41, no. 1, pp. 135–147, 1999.

[6] R. J. Clarke, "Relation between the Karhunen Loève and cosine transforms," *IEE Proceedings, Part F*, vol. 128, no. 6, pp. 359–360, Nov. 1981.

[7] D. L. Donoho, M. Vetterli, R. A. DeVore, and I. Daubechies, "Data compression and harmonic analysis," *IEEE Trans. Inform. Theory*, vol. 44, no. 6, pp. 2435–2476, 1998, invited paper.

[8] S. W. Hong, Y. H. Chan, and W. C. Siu, "A new approach for real time reduction of blocking effect," *Signal Processing*, vol. 65, no. 3, pp. 337–346, 1998.

[9] J. Luo, C. W. Chen, K. J. Parker, and T. S. Huang, "Artifact reduction in low bit rate DCT-based image compression," *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1363–1368, 1996.

[10] Q. Lu, J. Luo, and R. L. Joshi, "Image processing method for reducing noise and blocking artifact in a digital image," US Patent, No. 6,636,645, Oct. 21, 2003.

[11] Y. Kawasaka, "Block distortion corrector and image signal expander," US Patent, No. 5,949,917, Sep. 7, 1999.

[12] D. Zhang and Z. Wang, "Image information restoration based on long-range correlation," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 12, no. 5, pp. 331–341, 2002.

[13] G. Lakhani, "Improved image reproduction from dc components," *Opt. Eng.*, vol. 35, no. 12, pp. 3449–3452, 1996.

[14] ——, "Improved equations for JPEG's blocking artifacts reduction approach," *IEEE Trans. Circ. Syst. Vid.*, vol. 7, no. 6, pp. 930–934, 1997.

[15] V. T. Kieu and D. T. Nguyen, "Surface fitting approach for reducing blocking artifacts in low bitrate DCT decoded images," in *Proc. 2001 International Conf. Image Proc.* IEEE, 2001, pp. 150–153.

[16] N. Saito and J.-F. Remy, "A new local sine transform without overlaps: A combination of computational harmonic analysis and PDE," in *Wavelets: Applications in Signal and Image Processing X*, M. A. Unser, A. Aldroubi, and A. F. Laine, Eds., vol. Proc. SPIE 5207, 2003, pp. 495–506.

[17] ——, "The polyharmonic local sine transform: A new tool for local image analysis and synthesis without edge effect," *Applied and Computational Harmonic Analysis*, vol. 20, pp. 41–73, 2006.

[18] G. A. Triantafyllidis, D. Tzovaras, and M. G. Strintzis, "Blocking artifact detection and reduction in compressed data," *IEEE Trans. Circ. Syst. Vid.*, vol. 12, no. 10, pp. 877–890, 2002.

[19] C. Lanczos, *Applied Analysis*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1956, republished by Dover Publications, Inc. in 1988.

[20] R. R. Coifman and Y. Meyer, "Remarques sur l'analyse de Fourier à fenêtre," *Comptes Rendus Acad. Sci. Paris, Série I*, vol. 312, pp. 259–261, 1991.

[21] H. S. Malvar, "The LOT: transform coding without blocking effects," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, pp. 553–559, 1989.

[22] ——, "Lapped transforms for efficient transform/subband coding," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, pp. 969–978, 1990.

[23] A. Averbuch, M. Israeli, and L. Vozovoi, "A fast Poisson solver of arbitrary order accuracy in rectangular regions," *SIAM J. Sci. Comput.*, vol. 19, no. 3, pp. 933–952, 1998.

[24] E. Braverman, M. Israeli, A. Averbuch, and L. Vozovoi, "A fast 3D Poisson solver of arbitrary order accuracy," *J. Comput. Phys.*, vol. 144, pp. 109–136, 1998.

[25] G. B. Folland, *Introduction to Partial Differential Equations*, 2nd ed. Princeton Univ. Press, 1995.

[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error measurement to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–613, 2004.

[27] S. Minami and A. Zakhor, "An optimization approach for removing blocking effects in transform coding," *IEEE Trans. Circ. Syst. Vid.*, vol. 5, no. 2, pp. 74–82, 1995.

[28] G. Lakhani and N. Zhong, "Derivation of prediction equations for blocking effect reduction," *IEEE Trans. Circ. Syst. Vid.*, vol. 9, no. 3, pp. 415–418, 1999.